UM ALGORITMO BASEADO EM GRASP E BUSCA TABU APLICADO À RESOLUÇÃO DO PROBLEMA DE ROTEAMENTO DE VEÍCULOS COM JANELA DE TEMPO

Tatiana Alves Costa
Centro Federal de Educação Tecnológica de Minas Gerais
Av. Amazonas 7675 Nova Gameleira 30510-000 Belo Horizonte MG
tatiana@iceb.ufop.br

Sérgio Ricardo de Souza
Centro Federal de Educação Tecnológica de Minas Gerais
Av. Amazonas 7675 Nova Gameleira 30510-000 Belo Horizonte MG
sergio@dppg.cefetmg.br

Marcone Jamilson Freitas Souza
Universidade Federal de Ouro Preto
Departamento de Computação
marcone@iceb.ufop.br

Resumo

Este trabalho apresenta uma metodologia, baseada nas metaheurísticas GRASP e Busca Tabu, para resolver o Problema de Roteamento de Veículos com Janelas de Tempo. Esse problema tem como objetivo determinar as rotas de custo mínimo para uma frota de veículos de mesma capacidade, atendendo à demanda de um conjunto de clientes, para os quais o atendimento somente é possível dentro de um intervalo de tempo determinado, chamado janela de tempo. A metodologia desenvolvida parte de uma solução inicial gerada via um procedimento construtivo parcialmente guloso, baseado na heurística de Clarke e Wright. Posteriormente, essa solução é refinada pela metaheurística Busca Tabu. O método proposto foi testado em um conjunto de 40 problemas-teste, obtendo um bom desempenho e, em algumas instâncias, melhores resultados que os encontrados na literatura.
Palavras-Chaves: Roteamento de Veículos; Metaheurísticas; GRASP; Busca Tabu.

Abstract

This work presents an algorithm based on GRASP and Tabu Search metaheuristics for solving the Vehicle Routing Problem with Time Windows. The goal of this problem is to find a set of routes with minimum cost (in terms of distance traveled) for a fleet of identical vehicles based at the depot, obeying the capacity of the vehicles and the demand of a set of costumers and its predefined time windows. The proposed method uses an adaptation of the classic Clark and Wright heuristic for construct an initial solution to the problem. The refinement is made by the Tabu Search metaheuristic. The proposed method was applied to a 40 benchmarks and a good performance was obtained and, in some instances, better results than the known results from literature.
Keywords: Vehicle Routing, Metaheuristics, GRASP, Tabu Search.

1. INTRODUÇÃO

O Problema de Roteamento de Veículos com Janela de Tempo (PRVJT) é uma variante do problema clássico de roteamento de veículos, na qual o serviço de atendimento em cada consumidor deve ser iniciado dentro de um intervalo de tempo denominado janela de

tempo. O PRVJT pertence à classe de problemas NP-difícil, o que significa dizer que não existem algoritmos eficientes que encontrem a solução ótima para esse problema em um tempo computacional viável. Isso torna a utilização de métodos exatos bastante restrita e justifica o crescente uso de técnicas heurísticas e metaheurísticas aplicadas na resolução de problemas desse tipo.

O PRVJT pode ser definido como: seja $G=\{V, A\}$ um grafo, no qual $V=\{v_0, v_1, ..., v_n\}$ é o conjunto de vértices e $A=\{(v_i,v_j) : v_i, v_j \in V, i \neq j\}$ é o conjunto de arcos. O vértice v_0 representa o depósito (D), enquanto os demais n vértices correspondem aos clientes. Ao conjunto A estão associadas uma matriz de custos (c_{ij}) e uma matriz de tempos de viagens (t_{ij}). Se estas matrizes são simétricas, pode-se definir o PRVJT em um grafo não direcionado $G=\{V, E\}$, no qual $E=\{(v_i,v_j) : v_i, v_j \in V, i \neq j\}$ é um conjunto de arestas. Cada cliente tem uma demanda não-negativa m_i e um tempo de atendimento t_i. Uma frota de veículos idênticos de capacidade Q é atribuída ao depósito. O número de veículos pode ser conhecido com antecedência ou tratado como uma variável de decisão. Neste trabalho, considera-se que o número de veículos é ilimitado. O PRVJT consiste em designar um conjunto de rotas tal que:

- cada rota inicia e termina no depósito;
- cada cliente é visitado exatamente uma vez por exatamente um veículo;
- a demanda total de cada rota não exceda a capacidade Q de cada veículo;
- a janela de tempo $[e_i, l_i]$ deve ser respeitada, ou seja, um cliente não pode ser atendido antes do tempo e_i e nem após o tempo l_i; e
- o custo total da rota seja minimizado.

Encontrar uma solução de boa qualidade para o PRVJT é de fundamental importância, uma vez que, segundo [1], o custo de transporte corresponde de um terço a dois terços dos custos logísticos de uma empresa.

Devido à importância prática do PRVJT e o desafio de se resolvê-lo eficientemente, várias técnicas de solução têm sido relatadas na literatura.

Uma das primeiras abordagens heurísticas para resolução do PRV foi proposta em [2], sendo conhecida como heurística de Clarke e Wright. Trata-se de uma heurística gulosa para resolução do PRV, baseada no conceito de economia. O método parte de uma solução inicial, na qual cada consumidor é atribuído a um veículo e, a cada passo, o método tenta unir duas rotas, de forma a produzir a melhor economia no custo total de viagem, respeitando-se as restrições do problema. O método pára quando não existem mais uniões possíveis.

Em [8] é apresentada uma adaptação da heurística de Clarke e Wright ao PRV com frota heterogênea, no qual os veículos possuem diferentes capacidades. Várias fórmulas foram desenvolvidas na tentativa de incorporar, à fórmula original de economia proposta por Clarke e Wright, os custos fixos relativos a cada tipo de veículo que poderia ser utilizado.

Em [11], por sua vez, baseados em [8], são inseridas no problema restrições de janela de tempo pertinentes a cada cliente, tratando-se, assim o PRV com frota heterogênea e janelas de tempo.

Devido à grande complexidade do PRV e de suas variantes, os métodos metaheurísticos têm sido os mais utilizados para abordar esses problemas.

A metaheurística Busca Tabu (BT) é aplicada ao PRV com frota heterogênea em [4]. Nesse trabalho, um procedimento de memória adaptativa é usado para gerar soluções iniciais para a metaheurística BT. As inserções dos clientes às rotas são baseadas na heurística GENIUS, apresentada em [5], com o objetivo de resolver o Problema do Caixeiro Viajante.

O mesmo algoritmo mostrado em [4] é aplicado em [13] ao PRV com frota heterogênea. Entretanto, o procedimento proposto difere do anterior na definição da estrutura de vizinhança que, em seu caso, é gerada através de trocas e inserções convencionais. Além disso, cada rota da solução construída é re-otimizada através da aplicação de um algoritmo exato proposto para o Problema do Caixeiro Viajante.

As metaheurísticas *Simulated Annealing*, Busca Tabu e Algoritmos Genéticos são aplicadas ao PRVJT em [14]. Um procedimento construtivo é utilizado para gerar soluções

inicias factíveis para o problema, atribuindo, a cada iteração, um cliente a uma rota, respeitando-se as restrições de capacidade do veículo e a janela de tempo de cada cliente. A estrutura de vizinhança é gerada por movimentos de dupla troca de clientes entre rotas.

No presente trabalho, é proposta uma metodologia, baseada nas metaheurísticas GRASP (proposta por Feo e Resende em [3]) e Busca Tabu, para resolver o Problema de Roteamento de Veículos com Janelas de Tempo. No método proposto, uma solução inicial é construída por meio de um procedimento parcialmente guloso, baseado na heurística de Clark e Wright, seguindo as idéias da fase de construção GRASP. O refinamento dessa solução é feito pelo método Busca Tabu. A hibridização com a metodologia GRASP advém da possibilidade de melhoria de desempenho do método de Busca Tabu, tendo em vista que este exige uma boa solução inicial.

O trabalho está organizado da seguinte forma: a seção 2 apresenta a heurística de Clarke e Wright, tomada como base do procedimento construtivo utilizado na metodologia proposta. As seções 3 e 4 descrevem, respectivamente, as metaheurísticas GRASP e Busca Tabu. A seção 5 apresenta, em detalhes, a metodologia proposta para a solução do PRVJT. Os resultados obtidos são apresentados e discutidos na seção 6. A seção 7 conclui o trabalho.

2. A HEURÍSTICA DE CLARKE E WRIGHT

Segundo [10], o algoritmo de economias de Clarke e Wright é, talvez, uma das heurísticas mais conhecidas e utilizadas em problemas de roteamento de veículos. Ele é geralmente aplicado em problemas nos quais o número de veículos é também uma variável de decisão, ou seja, além de minimizar o custo total de uma solução, deseja-se também, indiretamente, determinar o número mínimo de veículos necessários para atender a todos os consumidores. O método baseia-se na noção de economias, que pode ser definida como o custo da combinação, ou união, de duas subrotas existentes. Trata-se de uma heurística iterativa de construção, baseada numa função gulosa de inserção.

O procedimento se inicia com cada consumidor sendo servido por um veículo, constituindo, assim, rotas entre o depósito e cada um dos consumidores. Logo, serão criadas tantas rotas quanto forem o número de consumidores. Seja c_{ij} o custo de viagem partindo de um consumidor i a um consumidor j, podendo ser dado em termos de distância percorrida ou tempo de deslocamento. O próximo passo do procedimento é realizar todas a combinações possíveis entre duas rotas, de modo que um veículo possa ser eliminado e o custo de viagem reduzido. Isto é, deve-se calcular o valor S_{ij} (denominado economia) obtido pela união de duas rotas. O cálculo da economia é dado pela expressão $S_{ij} = c_{i0} + c_{0j} - c_{ij}$. As duas rotas que levarem à maior economia de combinação, respeitando-se as restrições de capacidade e tempo definidas pelo problema, são unidas. O método termina quando não for mais possível combinar rotas, de forma a produzir economia positiva, respeitando-se as restrições impostas.

Segundo [11], duas rotas contendo os clientes i e j podem ser combinadas, desde que i e j estejam na primeira ou na última posição de suas respectivas rotas e que a demanda total das rotas combinadas não ultrapasse a capacidade do veículo. Além disso, para problemas de roteamento com janelas de tempo, deve-se ainda observar se, após a união das duas rotas, as restrições de tempo de atendimento definidas pelos consumidores ainda serão respeitadas.

3. A METAHEURÍSTICA GRASP

A metaheurística GRASP (Procedimento de Busca Adaptativa Gulosa e Aleatória) foi proposta em [3]. Trata-se de um procedimento iterativo, constituído de duas fases: uma fase de construção, na qual uma solução para o problema é gerada, e uma fase de refinamento ou busca local, na qual a vizinhança da solução corrente é explorada em busca de um ótimo local. No procedimento GRASP, as fases de construção e refinamento são repetidas um determinado número de vezes e o procedimento retorna, como resultado, a melhor solução encontrada ao longo de todas as iterações realizadas.

Durante a fase de construção, uma solução é gerada iterativamente, elemento por

elemento. A cada passo desse processo iterativo, os elementos candidatos a serem inseridos na solução em construção são armazenados em uma lista de candidatos C e ordenados, de acordo com algum critério de ordenação pré-definido. Este processo de ordenação é baseado em uma função adaptativa gulosa $g : C \rightarrow \Re$, que estima o benefício da escolha de cada um dos elementos armazenados na lista de candidatos. O pseudo-código apresentado na Figura 1 ilustra a fase de construção do procedimento GRASP.

procedimento $Construcao(g(.), \alpha, s)$;
1 $s \leftarrow \emptyset$;
2 Inicialize o conjunto C de candidatos;
3 enquanto $(C \neq \emptyset)$ faça
4 $g(t_{min}) = \min\{g(t) \mid t \in C\}$;
5 $g(t_{max}) = \max\{g(t) \mid t \in C\}$;
6 $LCR = \{t \in C \mid g(t) \leq g(t_{min}) + \alpha(g(t_{max}) - g(t_{min}))\}$;
7 Selecione, aleatoriamente, um elemento $t \in LCR$;
8 $s \leftarrow s \cup \{t\}$;
9 Atualize o conjunto C de candidatos;
10 fim-enquanto;
11 Retorne s;
fim $Construcao$;

Figura 1: Procedimento construtivo do método GRASP.

O parâmetro α, definido no procedimento construtivo, assume valores dentro do intervalo *[0,1]* e é responsável por controlar o nível de gulosidade e aleatoriedade do procedimento. Um valor de $\alpha = 0$ faz com que o procedimento trabalhe de forma totalmente gulosa, ou seja, escolhendo sempre, para compor a solução, o elemento que irá produzir o maior benefício no valor da função objetivo. Já um valor de $\alpha = 1$ faz com que a escolha do próximo elemento a ser inserido seja realizada de forma totalmente aleatória.

Para refinar a solução e explorar o espaço de soluções do problema à procura de um ótimo local, o procedimento GRASP clássico utiliza um método de descida.

4. A METAHEURÍSTICA BUSCA TABU

A Metaheurística Busca Tabu foi proposta, independentemente, em [6] e em [9]. É um procedimento adaptativo, dotado de uma estrutura de memória, que aceita movimentos de piora para escapar de ótimos locais.

O procedimento parte de uma solução inicial qualquer e, a cada iteração do método, explora o espaço de busca da solução corrente, a procura do seu melhor vizinho. O melhor vizinho encontrado torna-se, então, a solução corrente, mesmo que esta piore o valor da melhor solução encontrada até o momento. Esse processo é repetido até que o critério de parada adotado pelo método seja satisfeito.

Ao aceitar como melhor vizinho soluções de piora, o procedimento consegue escapar de ótimos locais encontrados durante a solução do problema, mas pode, entretanto, gerar ciclos, fazendo com que o procedimento retorne a uma solução já gerada anteriormente. Para evitar a ciclagem, o método trabalha com uma lista, denominada lista tabu, na qual são armazenados os movimentos ditos tabus, ou seja, aqueles movimentos que, quando realizados, levam a uma solução já analisada anteriormente.

A lista tabu clássica contém os movimentos reversos aos últimos $|T|$ movimentos realizados (sendo $|T|$ é um parâmetro do método) e funciona como uma fila de tamanho fixo,

ou seja, quando um novo movimento é adicionado à lista, o mais antigo é retirado. O tamanho $|T|$ da lista tabu define por quantas iterações um movimento será considerado tabu.

A lista tabu pode ser uma solução para o problema da ciclagem, mas, por outro lado, também pode proibir movimentos que levariam à soluções ainda não analisadas. Para contornar essa situação, o método conta com uma função de aspiração, que retira, sob certas circunstâncias, o *status* tabu de um determinado movimento. O critério de aspiração mais utilizado pelas implementações de métodos BT é o de aceitar um movimento tabu se este levar a uma solução s' cujo valor seja melhor que o valor da melhor solução s^* encontrada até o momento.

Como critérios de parada para o método BT, duas regras são mais difundidas e utilizadas. Pela primeira regra, o procedimento é interrompido após um certo número de iterações sem melhora no valor da melhor solução encontrada até então. Pela segunda regra, o método termina quando o valor da melhor solução chega a um limite inferior já conhecido ou aceitável.

5. A METODOLOGIA PROPOSTA

Esta seção apresenta a metodologia desenvolvida para a resolução do PRVJT. Detalha-se, também, como uma solução para o problema é representada; quais os tipos de movimentos considerados para explorar o espaço de soluções; e qual a função de avaliação utilizada para guiar essa exploração.

5.1. REPRESENTAÇÃO DE UMA SOLUÇÃO INICIAL

Neste trabalho, uma solução para o PRVJT é representada pelo conjunto das rotas que a compõe e por um valor associado, que determina o custo total dessa solução, geralmente expresso em termos da distância total percorrida e/ou tempo gasto na execução das rotas envolvidas nesta solução. Esse valor é dado por uma função de avaliação, que é discutida em detalhes na seção seguinte. A Figura 2 apresenta um exemplo desta representação. A solução mostrada é composta de sete rotas e o custo total dessa solução é de 468.96 u.m. A cidade *0* (zero) representa o depósito.

0-12-0	0-7-13-0	0-5-10-11-20-8-0	0-21-18-17-16-19-0	0-4-1-3-9-0	0-15-2-6-0	0-14-0
Valor da Função de Avaliação: 468,96						

Figura 2: Exemplo de representação de uma solução para o PRVJT.

5.2. FUNÇÃO DE AVALIAÇÃO

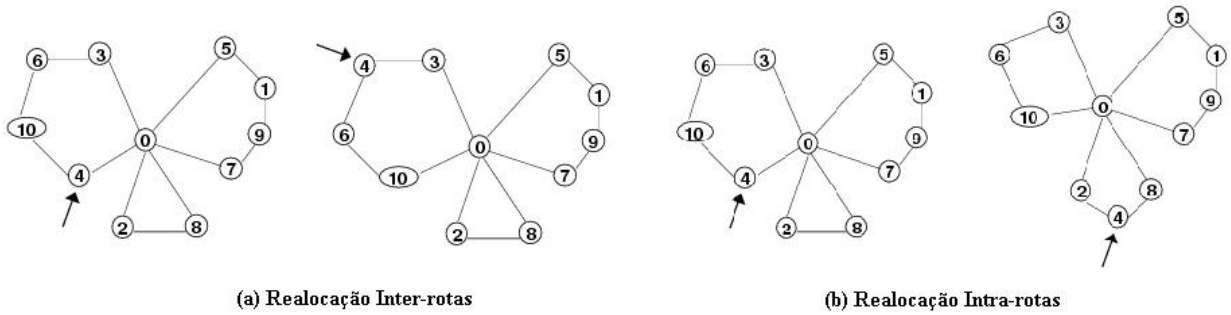
Foi adotada uma função de avaliação baseada em penalidades, que avalia cada rota referente a uma dada solução e penaliza eventuais violações às restrições de capacidade e janelas de tempo definidas pelo problema. A função de avaliação f utilizada é dada pela fórmula (1):

$$f(s) = \sum_{i,j \in A} d_{ij} + \beta \times E(s) + \gamma \times W(s) \qquad (1)$$

Nessa expressão, o termo $\sum_{i,j \in A} d_{ij}$ representa a função objetivo propriamente dita, ou seja, a soma das distâncias percorridas por todos os veículos da solução analisada. $E(s)$ equivale à soma dos excessos das capacidades em cada rota, caso exista, e $W(s)$ à soma de todas as violações às restrições de janelas de tempo, também caso existam. Os parâmetros β e γ são fatores de penalidades não negativos, determinados de forma auto-adaptativa, definidos, respectivamente, como a capacidade do veículo envolvido no roteamento e o maior tempo de viagem entre duas cidades do problema considerado.

5.3. TIPOS DE MOVIMENTOS E ESTRUTURA DE VIZINHANÇA

As metodologias implementadas neste trabalho utilizam o movimento de realocação para explorar o espaço de soluções do problema. Um movimento de realocação consiste em retirar uma cidade de uma rota qualquer e inseri-la em uma outra posição, que pode ser tanto dentro da mesma rota quanto em uma outra rota distinta. O movimento de realocação dentro de uma mesma rota é chamado de realocação intra-rota e está ilustrado na Figura 3(a). Esta figura mostra a cidade *4* sendo retirada da sua posição original e realocada após a cidade *6*, pertencente à mesma rota. Já o movimento de realocação envolvendo duas rotas distintas é denominado realocação inter-rotas e está representado na Figura 3(b). Neste exemplo, a cidade *4* é retirada da rota a qual pertence e realocada em uma outra rota, entre as cidades *2* e *8*.



(a) Realocação Inter-rotas (b) Realocação Intra-rotas

Figura 3: Movimentos de realocação.

A estrutura de vizinhança considerada neste trabalho combina os dois tipos de movimentos apresentados. Assim, nesta estrutura, um vizinho *s'* é obtido de uma solução *s* através da aplicação de movimentos de realocação intra-rotas ou inter-rotas.

5.4. GERAÇÃO DE UMA SOLUÇÃO INICIAL PARA O PROBLEMA

Neste trabalho, foram desenvolvidos dois procedimentos construtivos para a geração de uma solução inicial para o PRVJT, ambos baseados na heurística construtiva de Clarke e Wright. Estes procedimentos, denominados, respectivamente, Clarke e Wright Parcialmente Guloso (CWPG) e Clarke e Wright Parcialmente Guloso com Filtro (CWPGF), são descritos a seguir.

5.4.1. CLARKE E WRIGHT PARCIALMENTE GULOSO

A heurística de Clarke e Wright (CW), descrita na seção 2, trabalha com uma escolha gulosa para a função de avaliação, que faz com que o procedimento busque unir sempre, a cada iteração, rotas que produzam a maior economia no valor da função considerada. Isso torna o método determinístico, ou seja, quando aplicado diversas vezes à uma mesma instância do problema, a solução gerada será sempre a mesma. Neste trabalho, é proposta uma variante da heurística CW, denominada Clarke e Wright Parcialmente Guloso (CWPG), que adota uma escolha parcialmente gulosa para a função de avaliação, o que significa dizer que não necessariamente serão unidas as rotas que produzirem a maior economia, fazendo com este procedimento incorpore também a característica de gulosidade/aleatoriedade apresentada pela fase construtiva da metaheurística GRASP.

O procedimento trabalha com uma lista denominada Lista de Candidatos Restrita (LCR), que armazena algumas das melhores economias que podem ser geradas em cada iteração do método. A cada passo, uma dessas economias é escolhida aleatoriamente e as rotas são unidas, de forma a refletir, no valor da função de avaliação, a economia selecionada. O fator de aleatoriedade inserido no método faz com que, ao se aplicar repetidas vezes o CWPG a uma mesma instância de um PRVJT, diferentes soluções sejam geradas.

O método CWPG desenvolvido está ilustrado na Figura 4. Neste procedimento, o conjunto *C* de candidatos é formado por todas as combinações de rotas factíveis (que

satisfazem às restrições de capacidade e janelas de tempo impostas pelo problema) e pelas economias produzidas por essas combinações. O parâmetro α define o tamanho da lista LCR, sendo esta lista composta pelas α % combinações com as melhores economias armazenadas em *C*.

```
procedimento CWPG(α, s);
1  s ← rotas iniciais do método CW;
2  Calcule as economias de união de rotas S_ij, ∀i, j;
3  Ordene as economias em ordem decrescente;
4  Inicialize o conjunto C de candidatos;
5  enquanto (C ≠ ∅) faça
6      Considere uma lista de candidatos restrita (LRC), formada pelas α%
       combinações de melhores economias da lista C;
7      Selecione, aleatoriamente, uma combinação t ∈ LCR;
8      s ← s ⋃ {t};
9      Atualize o conjunto C de candidatos;
10 fim-enquanto;
11 Retorne a solução s gerada;
fim CWPG;
```

Figura 4: Procedimento Clarke e Wright Parcialmente Guloso.

5.4.2. CLARKE E WRIGHT PARCIALMENTE GULOSO COM FILTRO

Com o objetivo de tornar ainda melhor a qualidade das soluções geradas, foi desenvolvido o procedimento Clarke Wright Parcialmente Guloso com Filtro (CWPGF), no qual o método CWPG, apresentado na seção anterior, é executado um número determinado de vezes, e a melhor solução encontrada é retornada como resposta do método. A idéia deste procedimento baseia-se no fato de que o procedimento construtivo CWPG é relativamente barato computacionalmente. Assim, pode ser vantajoso aplicá-lo repetidas vezes, de forma a gerar soluções de melhor qualidade rapidamente. A Figura 5 ilustra o pseudo-código para esta metodologia. O parâmetro *MaxGrasp* indica o número de vezes que o procedimento CWPG será executado.

```
procedimento CWPGF(s, MaxGrasp);
1  f* ← +∞;
2  s* ← s;
3  para i = 1 até MaxGrasp faça
4      s = CWPG( );
5      se f(s) < f* então
6          s* ← s;
7          f* ← f(s);
8      fim-se;
9  fim para;
10 s ← s*;
11 Retorne s;
fim CWPGF;
```

Figura 5: Procedimento Clarke e Wright Parcialmente Guloso com Filtro.

5.5. GRASP COM FILTRO E BUSCA TABU APLICADO AO PRVJT

De acordo com a literatura, como regra geral, o processo de refinamento de uma solução é bastante caro computacionalmente, ao contrário da fase de construção, que requer

bem menos esforço computacional. Devido a isso, foi proposta a metodologia GraspFiltro. Nesse método, o processo de construção da solução inicial é realizado pelo procedimento CWPGF, descrito na seção 5.4.3, e o refinamento é feito através da metaheurística Busca Tabu. A Figura 6 ilustra o pseudo-código da metodologia proposta.

```
procedimento GraspFiltro(s, MaxGrasp);
1  s ← CWPGF( );
2  s ← BT( );
3  Retorne s;
fim GraspFiltro;
```

Figura 6: Método proposto

A metaheurística Busca Tabu implementada aqui é mesma descrita na seção 4, agora adaptada para ser aplicada ao PRVJT. As principais adaptações a que o método BT é submetido são listadas a seguir:

- O método trabalha com a estrutura de vizinhança definida na seção 5.3;
- A cada iteração, toda a vizinhança de uma dada solução é analisada e o melhor vizinho é retornado;
- A lista tabu implementada é dinâmica, o que significa dizer que seu tamanho é periodicamente alterado, variando entre um limite inferior $|T|_{min}$ e um limite superior $|T|_{max}$, definidos como parâmetros do método;
- A função de aspiração adotada é dada pelo valor da melhor solução encontrada até o momento, ou seja, se um movimento tabu leva a uma solução *s'* tal que $f(s') < f(s^*)$, então o movimento é realizado, mesmo sendo tabu, pois leva a uma solução ainda não considerada.

Um movimento tabu é gerado quando um movimento de realocação é realizado com uma determinada cidade. Considerando que uma cidade foi escolhida para ser realocada em uma outra posição qualquer, o movimento tabu a ser gerado deve proibir que esta cidade retorne novamente para a rota de onde foi retirada. Assim, um movimento tabu é definido por uma dupla <*Cidade Realocada, Rota de Origem*>, que proíbe que a cidade realocada retorne à rota a qual pertencia antes que o movimento de realocação fosse realizado.

O tamanho da lista tabu é definido de forma auto-adaptativa, segundo proposto em [7]. Assim, é definida como sendo um valor dentro do intervalo $[t_{min}, t_{max}]$, no qual $t_{min} = 0.9 \times N$ e $t_{max} = 1{,}1 \times N$, truncando-se o valor obtido, sendo N representando o número de cidades do problema considerado. O tamanho escolhido é mantido constante por $2 \times t_{max}$ iterações.

6. APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

O algoritmo proposto foi desenvolvido na linguagem C, utilizando o compilador C++ Builder 5.0 da Borland, e executado em um computador Athlon XP 2.6+, tendo 512 MB de memória RAM, sob plataforma Windows XP Pro.

O algoritmo foi testado e validado utilizando-se um conjunto de 40 instâncias teste de 100 consumidores, apresentadas em [12]. Essas instâncias se dividem em 2 grupos, a saber:

- grupo C: os clientes se encontram clusterizados, ou seja, estão distribuídos geograficamente em grupos de clientes próximos uns dos outros; e
- grupo R: os clientes são distribuídos aleatoriamente sem formar grupos e distantes uns dos outros.

Para a realização dos testes, os parâmetros utilizados nos procedimentos CWPG, CWPGF e BT estão apresentados na Tabela 1.

Tabela 1: Valores dos parâmetros utilizados no método *GraspFiltro*

Procedimento	Parâmetro
CWPG	$\alpha = 0.4$
CWPGF	*MaxGrasp = 10*
Busca Tabu	Nº de iterações sem melhora = 250

O algoritmo proposto foi executado dez vezes, cada qual partindo de uma semente diferente de números aleatórios.

Os melhores resultados da literatura, mencionados nas tabelas a seguir, advêm de duas fontes, cujo acesso se deu em 16/05/2005: (1) http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html e (2) http://w.cba.neu.edu/~msolomon.

Para cada instância analisada, são mostrados o desvio médio, calculado conforme fórmula (2), e o desvio da melhor solução, calculado conforme fórmula (3), apresentadas a seguir:

$$Desvio\ Médio = \frac{f(s^*) - \frac{1}{10}\left(\sum_{i=1}^{10} f(s_i)\right)}{f(s^*)} \tag{2}$$

sendo $f(s^*)$ o melhor resultado encontrado na literatura e $f(s_i)$ o resultado encontrado na *i*-ésima execução do algoritmo.

$$Desvio\ da\ Melhor\ Solução = \frac{f(s^*) - \min\{f(s_i);\ i = 1,...,10\}}{f(s^*)} \tag{3}$$

A Tabela 2 mostra o desempenho médio das metodologias CW, CWPG e GraspFiltro nos grupos de instâncias, isto é, a média dos desvios médios obtidos pela aplicação da fórmula (2) a todas as instâncias do grupo. No caso do método CW, como o resultado é determinístico, o procedimento é executado uma única vez para cada instância e o desvio médio corresponde ao desvio da melhor solução.

Tabela 2: Desempenho médio das metodologias CW, CWPG e GraspFiltro

Grupo de instâncias	CW	CWPG	GraspFiltro
C100	14,35%	20%	4,46%
R100	13,29%	22%	3,59%
Média	13,82%	21%	4,02%

A Tabela 3 mostra o melhor desempenho das metodologias CWPG e GraspFiltro nos grupos de instâncias, isto é, a média dos desvios das melhores soluções obtidas pela aplicação da fórmula (3) a todas as instâncias do grupo.

Tabela 3: Melhor desempenho das metodologias CWPGF e GraspFiltro

Grupo de instâncias	CWPG	GraspFiltro
C100	14,35%	1,35%
R100	10,26%	1,16%
Média	12,31%	1,26%

A Tabela 2 mostra que o procedimento CWPG é capaz de produzir soluções de qualidade média superiores ao produzido pela heurística clássica de Clark e Wright. Ela mostra, ainda, que o método GraspFiltro, o qual parte de uma solução inicial gerada pela aplicação repetida do procedimento CWPG, melhora ainda mais a qualidade da solução final.

A Tabela 3 mostra que o procedimento GraspFiltro produz soluções que desviam pouco das melhores soluções da literatura. Considerando que as melhores soluções da literatura foram obtidas por diferentes metodologias, conclui-se que o método GraspFiltro proposto é robusto.

A Tabela 4 mostra as características das soluções obtidas, bem como o desempenho da metodologia GraspFiltro em todas as instâncias analisadas. Nesta tabela, a coluna "DT" representa a distância total percorrida pelos veículos e "NV" o número de veículos da solução.

Tabela 4: Desempenho da metodologia GraspFiltro nas instâncias de 100 consumidores

Instância	Melhor Publicado		GraspFiltro				
	DT	NV	DT	NV	Desvio Médio	Desvio da Melhor Solução	Tempo médio (s)
C101-100	827,3 (2)	10	827,3	10	0,02%	0,00%	409,54
C102-100	827,3 (2)	10	836,0	10	6,29%	1,05%	432,73
C103-100	826,3 (2)	10	841,5	10	6,19%	1,83%	420,20
C104-100	822,9 (2)	10	836,5	10	5,56%	1,65%	431,51
C105-100	827,3 (2)	10	827,3	10	0,81%	0,00%	386,31
C106-100	827,3 (2)	10	827,3	10	0,91%	0,00%	412,29
C107-100	827,3 (2)	10	827,3	10	0,03%	0,00%	387,09
C108-100	827,3 (2)	10	827,3	10	0,05%	0,00%	426,98
C109-100	827,3 (2)	10	827,3	10	2,01%	0,00%	399,73
C201-100	589,1 (2)	3	589,1	3	1,94%	0,00%	596,60
C202-100	589,1 (2)	3	616,2	4	12,72%	4,60%	709,25
C203-100	588,7 (2)	3	615,8	4	12,35%	4,60%	698,28
C204-100	588,1 (2)	3	639,5	4	12,70%	8,74%	580,81
C205-100	586,4 (2)	3	586,4	3	5,40%	0,00%	661,60
C206-100	586,0 (2)	3	586,4	3	1,95%	0,06%	616,78
C207-100	585,8 (2)	3	587,9	3	3,95%	0,35%	608,84
C208-100	585,8 (2)	3	586,4	3	3,92%	0,10%	639,106
R101-100	1637,7 (2)	20	1719,8	21	6,28%	5,01%	408,12
R102-100	1466,6 (2)	18	1499,9	19	5,49%	2,27%	535,54
R103-100	1208,7 (2)	14	1243,4	15	6,95%	2,87%	395,22
R104-100	971,5 (2)	11	1018,1	12	7,88%	4,79%	523,84
R105-100	1355,3 (2)	15	1369,4	16	4,84%	1,04%	545,67
R106-100	1234,6 (2)	13	1271,4	15	5,83%	2,98%	597,78
R107-100	1064,6 (2)	11	1111,1	12	6,96%	4,36%	494,90
R108-100	960,88 (2)	9	979,9	11	4,37%	1,97%	462,85
R109-100	1146,9 (2)	13	1203,0	15	7,67%	4,89%	499,87
R110-100	1068,0 (2)	12	1130,5	13	7,74%	5,85%	424,29
R111-100	1048,7 (2)	12	1083,2	13	6,07%	3,28%	481,89%
R112-100	982,14 (2)	9	1004,7	13	4,31%	2,29%	375,59
R201-100	1143,2 (2)	8	1197,5	11	6,89%	4,74%	675,75
R202-100	1191,7 (2)	3	1113,0	11	-4,60%	-6,60%	641,95
R203-100	939,54 (2)	3	***917,2***	7	0,58%	***-2,37***	733,05
R204-100	825,52 (2)	2	***771,4***	5	***-4,25***	***-6,55***	597,48
R205-100	994,42 (2)	3	995,9	8	2,59%	0,14%	590,00
R206-100	906,14 (2)	3	917,4	6	4,15%	1,24%	722,2
R207-100	893,33 (2)	2	***842,3***	6	***-1,98%***	***-5,70%***	630,05
R208-100	726,75 (2)	2	732,6	4	6,37%	0,80%	400,02
R209-100	909,16 (2)	3	***897,5***	6	2,51%	***-1,28%***	560,38
R210-100	939,34 (2)	3	952,2	8	3,90%	1,37%	720,80
R211-100	892,71 (2)	2	***886,8***	6	***-8,10%***	***-0,66%***	340,07

5. CONCLUSÕES

Neste trabalho, é proposta uma metodologia, baseada nas metaheurísticas GRASP e Busca Tabu, para resolver o problema de roteamento de veículos com janela de tempo. A metodologia proposta, denominada GraspFiltro, parte de uma solução inicial gerada por um procedimento parcialmente guloso baseado na heurística de Clark e Wright. Esse procedimento é aplicado várias vezes, e a melhor das soluções encontradas é retornada como a solução inicial. Em seguida, essa solução é refinada pelo método de Busca Tabu.

Os resultados obtidos mostram que o método proposto produz soluções que desviam pouco das melhores soluções da literatura. Desse modo, considerando que essas foram obtidas por diferentes metodologias, conclui-se que o método GraspFiltro proposto é robusto.

8. REFERÊNCIAS BIBLIOGRÁFICAS

[1] BALLOU, R. (2001) *Gerenciamento da Cadeia de Suprimentos: Planejamento, Organização e Logística Empresarial*. São Paulo: Bookman, 2001.

[2] CLARK G.; WRIGHT, J. (1964) Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research*, v. 12, p. 568-581.

[3] FEO, T. A.; RESENDE, M. G. C. (1995) Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization*, v. 6, p. 109-133.

[4] GENDREAU, M.; LAPORTE, G.; MUSARAGANYI, C.; TAILLARD, E. D. (1999) A Tabu Search Heuristic for the Heterogeneous Fleet Vehicle Routing Problem. *Computers & Operations Research*, v. 26, p. 1153-1173.

[5] GENDREAU, M.; HERTZ, A.; LAPORTE, G. (1992) New Insertion and Postoptimization Procedures for the Traveling Salesman Problem. *Operations Research*, v. 40, p. 1086-1093.

[6] GLOVER, F. (1986) Future Paths for Integer programming and Links to Artificial Intelligence. *Computers & Operations Research*, v. 5, p. 549-553.

[7] GLOVER, F.; TAILLARD, E. D.; de Werra, D. (1993) A User's Guide to Tabu Search. Annals of Operations Research, v. 41, p. 3-28.

[8] GOLDEN, B.; ASSAD, A.; LEVY, L.; GHEYSENS, F. (1984) The Fleet Size and Mix Vehicle Routing. *Computers & Operations Research*. Grã-Bretanha, v. 11, n. 1, p. 49-66.

[9] HANSEN, P. (1986) The Steepest Ascent Mildest Descent Heuristic for Combinatorial Programming. Congress on Numerical Methods in Combinatorial Optimization, Capri, Itália.

[10] LAPORTE, G.; GENDREAU, M.; POTVIN, J. Y.; SEMET, F.(2000) Classical and Modern Heuristics for the Vehicle Routing Problem. *International Transactions in Operational Research*, v. 7, p. 285-300.

[11] LIU, F.; SHEN, S. Y. (1999) A Metohd for Vehicle Routing Problem with Multiple Vehicle Types and Time Windows. Proc. Natl. Sci. Counc.. Hsinchu, Taiwan, v. 23, p. 526-536.

[12] SOLOMON, M. M.; Algorithms for Vehicle Routing and Scheduling Problems with Time Windows Constraints. *European Journal of Operational Research*, v. 35, p. 254-266.

[13] TAILARD, É. (1993) Parallel iterative search methods for vehicle routing problems. *Networks*, v. 23, p. 661-673.

[14] TAN, K.C.; LEE, L. H.; ZHU, Q. L.; OU, K. (2001) Heuristic methods for vehicle routing problem with time windows. *Artificial Intelligence in Engineering*, v 15, p. 281-295.