PROBLEMA DE ROTEAMENTO DE VEÍCULOS ELÉTRICOS COM FROTA HETEROGÊNEA, ESTAÇÕES DE RECARGA E JANELAS DE TEMPO

Marcelo Edivan Freitas Santos[1]

[1]Graduando em Computação – Universidade Federal de Ouro Preto (UFOP)
Campus Universitário, Morro do Cruzeiro, CEP 35.400-000, Ouro Preto (MG), Brasil

edivanmarcelo@gmail.com

Resumo. *Este artigo tem seu foco em uma subclasse do problema de roteamento de veículos (PRV). No problema tratado além de ser considerado todos os aspectos do PRV convencional, novas características são incorporadas; a frota é heterogênea e composta por veículos elétricos, que podem ser reabastecidos durante a rota, e os clientes devem ser atendidos dentro de uma janela de tempo. A adição dessas novas características gera novas restrições o que acaba tornando o problema mais complexo. Para sua resolução, propõe-se um algoritmo heurístico Multi-Start que consiste em criar várias soluções iniciais diferentes e aperfeiçoá-las combinando procedimentos heurísticos como VNS e VND. Quando uma nova solução inicial é gerada pela heurística, ela é passada para ser refinada pelo algoritmo VNS, que tem como procedimento de busca local o VND, e caso essa solução retorne com uma melhoria em sua função objetiva a melhor solução é atualizada até que outra tome o seu lugar. Os resultados encontrados em alguns casos são de boa qualidade, mas no geral mostram que ainda há muito que ser feito para que se tornem mais efetivos.*

PALAVRAS-CHAVE: *Roteamento de Veículos, VND, VNS, Multi-Start, Veículos Elétricos, Janelas de Tempo, Frota Heterogênea.*

Abstract. *This article focuses on a subclass of the vehicles routing problems (VRP). In the problem dealt, besides being considered all aspects of the conventional VRP, new features are incorporated; the fleet is heterogeneous and composed of electric vehicles, which can be refueled during the route, and the clients must be served within a time window. The addition of these new features creates new constraints, which makes the problem more complex. For its resolution, is proposed a multi-start heuristic algorithm, consisting of create several different initial solutions and perfect them by combining heuristic procedures, such as VNS and VND. When a new initial solution is generated by the heuristic, it is passed to be refined by the VNS algorithm, which has the VND as local search procedure, and if this solution returns with an improvement in its objective function the best solution is updated until another takes its place. In same cases, the results found have good quality, but overall show that much remains to be done to make them more effective.*

KEYWORDS: *Vehicles routing, VND, VNS, Multi-Start, Electric Fleet, Time Windows, Fleet Size and Mix.*

1 INTRODUÇÃO

Estudos são realizados na área de Pesquisa Operacional com intuito de otimizar as rotas de distribuição em um sistema de logística e transporte trazendo benefícios como a diminuição do custo operacional das empresas e os impactos negativos dos sistemas de distribuição, como engarrafamento e poluição. E foi justamente para resolver o problema da poluição (causada pelo sistema de logística e transporte) que nasceu uma variante do Problema de Roteamento de Veículos (PRV), o qual é estudado neste trabalho, onde é feito o uso de uma frota heterogênea de veículos elétricos.

O Problema de Roteamento de Veículos Elétricos com Frota Heterogênea, Estações de Recarga e Janelas de Tempo (The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations) ou apenas E-FSMFTW, trata-se de tentar minimizar o custo de atendimento de clientes com janelas de tempo fazendo o uso de uma frota heterogênea de veículos elétricos.

O presente trabalho trata-se de efetuar a resolução do problema usando um algoritmo Multi-Start atrelado ao GVNS e ao VND, a fim de comparação dos resultados. Primeiramente é usado o algoritmo Multi-Start para geração da solução inicial, a partir daí a solução é refinada por meio do GVNS, nesta fase a ordem de movimentos para explorar o espaço de busca é embaralhada a cada entrada no GVNS para que as mais diversas combinações de movimentos seja explorada no decorrer da execução do método. O procedimento de busca local usado no GVNS é o VND onde mais uma vez a ordem de movimentos é embaralhada; quando o movimento é a aplicado na solução corrente, essa é refinada com o método de busca local first improvement e caso haja uma melhora a melhor solução é alterada, voltando ao GVNS e a partir daí ao Multi-Start onde todo o processo se reinicia até que o critério de parada seja atendido.

O restante do trabalho está organizado da seguinte forma. Na seção 2 o problema apresentado é caracterizado. Na seção 3 é feita a apresentação dos algoritmos heurísticos para resolução do E-FSMFTW. Os resultados computacionais são apresentados na seção 5 e, por fim, na seção 6 conclui-se o trabalho e as propostas a serem exploradas em trabalhos futuros são apontadas.

2 CARACTERIZAÇÃO DO PROBLEMA

O E-FSMFTW possui as seguintes características: (a) todos os clientes C devem ser atendidos apenas uma vez por um único veículo; (b) cada cliente i possui um tempo de atendimento S_i. O início do atendimento desse cliente i é delimitado pela janela $[E_i, T_i]$, em que E_i e T_i indicam, respectivamente, o instante de tempo inicial e final que se é possível começar o atendimento; (c) a demanda p_i do cliente i deve ser satisfeita ao se realizar o atendimento do mesmo; (d) os clientes podem ser atendidos por até k tipos de veículos diferentes, com custos de aquisição f^k distintos; (e) as capacidades de carregamento Q^k do veículo k não deve ser ultrapassada; (f) o total da carga da bateria Y^k do veículo k têm que ser maior que zero, para isso são usadas estações de recargas F, que são visitadas sempre que necessário; (g) todas as rotas começam e terminam no depósito $depot$ que também tem janela de tempo, a qual determina o tempo de duração máxima de cada rota; (h) não é permitido a um veículo k atender dois clientes C_i e C_j ao mesmo tempo.

O E-FSMFTW consiste em determinar boas rotas de atendimento para um determinado grupo de clientes, respeitando suas respectivas janelas de tempo, usando uma frota heterogênea de veículos elétricos de modo que o custo seja aceitável assim como o tempo de processamento gasto para encontrar as mesmas.

O objetivo é minimizar o custo de aquisição dos veículos usados para realizar as rotas somadas ao custo de atender os clientes.

3 METODOLOGIA

3.1 Representação da solução

A solução é representada como uma matriz M de tamanho $n\ X\ m$, onde o número de colunas m de cada linha é variável, acompanhado de um vetor v de tamanho n onde cada posição diz respeito ao veículo usado para realizar a rota. Cada linha completa M_i da matriz representa uma rota de atendimento, as posições da coluna M_{ij} dizem respeito a ordem de atendimento, podendo variar entre três tipos, c = Cliente, e = Estação, d = Depósito. Exemplo a seguir:

$M =$ $\{d_0, c_3, c_5, e_0, c_2, d_0\}$ $v = \{v_1, v_2\}$
$\{d_0, c_1, c_0, c_4, e_1, c_6, d_0\}$

3.2 Vizinhança

O espaço de soluções é explorado pelos 4 seguintes movimentos divididos em dois tipos:

3.2.1 Movimentos Intrarrotas

Realocação: Um cliente é removido e reinserido em outra posição da rota.

$M =$ $\{d_0, c_5, e_0, c_3, c_2, d_0\}$ $v = \{v_1, v_2\}$
$\{d_0, c_1, c_0, c_4, e_1, c_6, d_0\}$

O cliente c_3 passou da posição 2 para posição 4 da rota 1.

Troca: É feito uma permutação entre dois clientes da rota.

$M =$ $\{d_0, c_3, c_5, e_0, c_2, d_0\}$ $v = \{v_1, v_2\}$
$\{d_0, c_1, c_4, c_0, e_1, c_6, d_0\}$

Os clientes c_0 e c_4 da rota 2 trocaram de posição.

3.2.2 Movimentos Inter-rotas

Realocação: Um cliente é removido de uma rota e reinserido em outra rota.

$M =$ $\{d_0, c_3, c_5, e_0, c_0, c_2, d_0\}$ $v = \{v_1, v_2\}$
$\{d_0, c_1, c_4, e_1, c_6, d_0\}$

O cliente c_0 passou da posição 3 da rota 2 para posição 5 da rota 1.

Troca: É feita a permutação entre o cliente i de uma rota e o cliente j de outra.

$M =$ $\{d_0, c_3, c_4, e_0, c_2, d_0\}$ $v = \{v_1, v_2\}$
$\{d_0, c_1, c_0, c_5, e_1, c_6, d_0\}$

Os clientes c_4 e c_5 da rota trocaram de posição.

3.3 Função de Avaliação

A solução é avaliada pela seguinte equação que deve ter seu valor minimizado.

$$min \quad \sum_{k \in V} \sum_{j \in V} f^k x^k_{0j} \quad + \quad \sum_{k \in v} \sum_{i \in N,\, j \in N,\, i \neq j} c^k_{ij} x^k_{ij} \quad + \quad P \tag{1}$$

Onde temos os índices,

k : que representa o índice do veículo que faz a rota.

i e j : que representam a sequência da visita do cliente/estação/deposito dentro da matriz.

E as variáveis,

V : Conjunto dos tipos de veículos.

N : Conjunto de nós, união do conjunto de depósitos, clientes e estações de recarga

f^k : variável que diz respeito ao custo fixo (aquisição do veículo) do veículo k.

x_{ij}^k : é uma variável binaria que assume valor 1 se o veículo k foi usado para fazer a visita do nó i para o nó j e 0 caso contrário.

c_{ij}^k : representa o custo para que o veículo k vá do nó i para o nó j.

P : representa um custo adicional por penalização caso algum cliente seja atendido após o término da sua janela de tempo, caso o veículo chegue antes do início da janela não há penalização apenas é adicionado o tempo de espera.

3.4 Algoritmos Propostos

Os resultados foram obtidos a partir da junção entre dois algoritmos heuristicos, onde o GVNS se tornou o procedimento de busca local do algoritmo Multi-Start. Já no GVNS o procedimento de busca local é o VND.

3.4.1 Algoritmo Multi-Start

O primeiro dos três algoritmos usados para resolver o problema é descrito a seguir no Algoritmo 1.

Algoritmo 1: Multi-Start

```
1  início
2      f* ← ∞
3      enquanto (Critério de parada não atendido) faça
4          s ← ConstruaSolucao( );
5          s ← GVNS(s);
6          se (f(s) < f(s*)) então
7              s* ← s;
8              f* ← f(s);
9          fim-se;
10     Fim-enquanto;
11     s ← s*;
12     Retorne s;
13 fim
```

Na linha 1 é adicionado o valor ∞ a variável f^* associada a s* que por sua vez guarda a melhor solução encontrada, isso é feito para que na primeira execução do laço de repetição, a solução gerada tome o lugar da melhor solução. Na linha 4 uma nova solução é gerada e armazenada na variável s; logo após na linha 5 a solução s é refinada por meio do GVNS. Então é feita a comparação do valor da função objetiva da melhor solução gerada e da solução corrente, linha 6; caso a solução corrente seja melhor, a melhor solução é substituída; caso contrário é dado continuidade ao laço de repetição até que o critério de parada seja atendido. Ao sair do laço a variável s recebe a melhor solução e é retornada.

3.4.2 Algoritmo GVNS

O segundo algoritmo, GVNS, é apresentado a seguir no Algoritmo 2.

Algoritmo 2: GVNS

```
1  início
2      s ← s0;
3      embaralha movs[r]
4      enquanto (Critério de parada não atendido) faça
5          k ← 1;
6          enquanto (k < r) faça
7              gere um vizinho qualquer s'∈N^(mov[k]);
8              s" ← VND(s');
9              se ( f(s") < f(s) )
10                 então s ← s"; k ← 1;
11                 senão k ← k + 1;
12             fim-se;
13         fim-enquanto;
14     fim-enquanto;
15     Retorne s;
16 fim
```

Na linha 2 a variável s recebe à solução inicial s_0; já na linha 3 o vetor movs[r] que guarda as estruturas de vizinhanças é embaralhado, onde r é o número de estruturas de vizinhanças e k é uma variável inteira utilizada para caminhar dentro das estruturas de vizinhanças; na linha 7 um vizinho qualquer é gerado a partir do movimento k e então esse vizinho é refinado por meio do VND e atribuído a variável s", caso a nova solução seja melhor que a solução guardada em s, s é atualizada e k volta ao valor 1 referenciando o primeiro movimento; caso contrário k é somado em 1 para que a próxima estrutura de vizinhança seja usada. É repetido até que o critério de parada seja atendido. Por fim a solução s é retornada.

3.4.3 Algoritmo VND

Por fim o último algoritmo usado para resolução do problema é descrito a seguir no Algoritmo 3.

Algoritmo 3: VND

```
1  início
2      s ← s0;
3      embaralha movs[r]
4      enquanto (Critério de parada não atendido) faça
5          k ← 1;
6          enquanto (k < r) faça
7              encontre o primeiro melhor vizinho s'∈N^(mov[k])(s);
8
9              se ( f(s') < f(s) )
10                 então s ← s'; k ← 1;
11                 senão k ← k + 1;
12             fim-se;
13         fim-enquanto;
14     fim-enquanto;
15     Retorne s;
16 fim
```

Na linha 2 a variável s_0 com uma solução inicial é atribuída a variável s; já na linha 3 o vetor movs[r] o qual guarda os movimentos que geram as estruturas de vizinhanças é embaralhado, onde r

é o número de estruturas de vizinhanças e k é uma variável inteira utilizada para caminhar dentro do espaço dessas estruturas; na linha 7 o primeiro melhor vizinho é gerado a partir do movimento k e armazenado na variável s', caso a nova solução seja melhor que a solução guardada em s, s é atualizada e k volta ao valor 1 referenciando o primeiro movimento; caso contrário k é somado em 1 para que a próxima estrutura de vizinhança seja usada. O laço mais externo é repetido até que o critério de parada seja atendido. Por fim a solução s é retornada.

4 SOLUÇÃO INICIAL

A solução inicial é gerada pelo algoritmo ConstruaSolucao, descrito a seguir no Algoritmo 4.

Algoritmo 4: ConstruaSolucao

```
1  início
2      s ← vazio;
3      V ← vazio ;
4      enquanto (ainda há clientes não atendidos) faça
5          v ← qualquer veículo entre os k tipos;
6          V ← V ∪ {v};
7          enquanto (capacidade do veículo v não for atingida) faça
8              depot ← deposito qualquer ;
9              s ← s ∪ {depot};
10             se (veículo v sem carga) então
11                 s ← s ∪ {estação mais próxima do ultimo cliente inserido};
12             fim-se;
13             s ← s ∪ {ProcuraClienteMaisProximo( )};
14         fim-enquanto;
15         s ← s ∪ {depot};
16     fim-enquanto;
17     Retorne s;
18 fim
```

Primeiramente, na linha 5, um veículo v é escolhido e então inserido no vetor V que armazena para cada posição o veículo usado na rota; então até que a capacidade desse veículo seja atingida, um depósito é inserido na rota (no caso do problema abordado aqui há apenas um depósito), linha 9. Na linha 10, caso o veículo esteja sem carga uma estação de recarga é inserida na rota para o abastecimento; logo após é feito a inserção do cliente por meio do método ProcuraClienteMaisProximo( ); quando a capacidade do veículo é atingida a rota acaba e o depósito é inserido ao final, o que marca o fim da rota. Tudo isso é repetido até que todos os clientes sejam atendidos.

O método ProcuraClienteMaisProximo( ) é feito como segue, vide algoritmo 5.

Algoritmo 5: ProcuraClienteMaisProximo

```
1 início
2     c' ← o ultimo nó inserido em s;
3
4     enquanto (probabilidade satisfeita) faça
5         c ← cliente mais próximo de c' ;
6
7     fim-enquanto;
8     Retorne c;
9 fim
```

O método consiste em basicamente selecionar um novo cliente, mais próximo ao último inserido na rota, sempre que uma probabilidade for aceita, isso é feito para que as soluções criadas por ele sejam distintas e ainda assim seguindo o conceito do o cliente mais próximo.

5 Resultados computacionais

Para realizar os experimentos foram usados um conjunto de instâncias presentes na literatura com 100 clientes divididos em três grupos: A, B e C. Cada conjunto de instâncias possuem três tipos diferentes: tipo C, onde grupos de clientes estão localizados mais próximos; Já no tipo R a localização dos clientes é feita de forma randômica; e no tipo RC é feita uma união entre os tipos C e R.

Os algoritmos descritos foram desenvolvidos na linguagem C++, utilizando-se o IDE NetBeans 8.2. Todos os experimentos computacionais foram executados em um notebook com Intel Core i5 – 5200U 2,2 GHZ com 8 GB de memória RAM e sistema operacional Windows 10 64 bits. O critério de parada usado foi o número de iterações sem melhoras o qual assumiu valor 15; que também foi o número de clientes usados nos testes.

A media dos resultados dos testes é apresentado a seguir na Tabela 1.

Tipo	*Instância*	*Clientes*	*Valor Função*	*Tempo(s)*
A	c103_21	15	**980.81**	10
	c106_21	15	**1351.66**	4.1
	r102_21	15	**1696.03**	6.52
	r105_21	15	**1628.50**	2.1
	rc103_21	15	**1265.70**	4.6
	rc108_21	15	**1435.15**	2.3
B	c103_21	15	**425.80**	7.28
	c106_21	15	**417.51**	4.91
	r102_21	15	**980.01**	3.6
	r105_21	15	**1024.05**	3.47
	rc103_21	15	**1207.82**	0.9
	rc108_21	15	**1120.05**	0.61
C	c103_21	15	**717.8**	1.28
	c106_21	15	**486.01**	1.6
	r102_21	15	**945.53**	3.25
	r105_21	15	**1104.3**	2.97
	rc103_21	15	**871.7**	0.95
	rc108_21	15	**755.8**	1.25

Tabela 1: *Resultados Computacionais*

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho teve seu foco em resolver o problema de roteamento de veículos elétricos com frota heterogênea, estações de recarga e janelas de tempo, onde o objetivo é minimizar o custo de aquisição dos veículos usados para realizar as rotas somados ao custo de atender os clientes de forma que suas respectivas janelas de tempo sejam respeitadas bem como a capacidade e carga de cada um dos veículos usados.

Foi proposto uma combinação de três algoritmos heurísticos para se fazer a resolução do problema, a começar pelo Multi-Start em que várias soluções são construídas, a cada solução criada ela é enviada ao algoritmo GVNS para ser refinada que por sua vez altera a solução para um vizinho que é escolhido pelo movimento atual a partir do vetor de movimentos movs[r]; isso é feito para que a solução não fique presa em ótimos locais e assim o espaço de busca possa ser melhor explorado. O vizinho gerado no GNVS é mandado para ser refinado por meio do terceiro algoritmo usado, o VND, onde novamente as estruturas de vizinhanças são empregadas, com a diferença de que o vizinho escolhido é o primeiro a apresentar uma melhora da função objetiva.

Para trabalhos futuros o foco é em otimizar esse trabalho criando novos movimentos para explorar o espaço de busca como, por exemplo, movimentos para alteração dos veículos e estações de recarga, além de reorganizar o código fonte para que a execução do mesmo se torne mais rápida, bem como tornar o código fonte da solução inicial mais simples.

AGRADECIMENTOS

O autor agradece à FAPEMIG pelo apoio recebido.

REFERÊNCIAS

Hiermann, Gerhard; Puchinger, Jakob; Ropke, Stefan; Hartl, Richard F. **The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations**. European Journal of Operational Research 252 (2016) 995–1018, 2016.

Notas de aula do professor Marcone Jamilson Freitas Souza. **Variable Neighborhood Descent (VND) e Variable Neighborhood Search(VNS).**

Notas de aula do professor Marcone Jamilson e Puca Huachi. **Multi-Start**.