Um GRASP-RVND para o Problema de Roteamento de Veículo Elétrico com Janelas de Tempo e Recargas Parciais

Filipe Augusto de Oliveira Leoncio, Luciana Brugiolo Gonçalves, Lorenza Leão Oliveira Moreno, Stênio Sã Rosário Furtado Soares
Departamento de Ciência da Computação
Universidade Federal de Juiz de Fora
filipeaugusto@ice.ufjf.br, lbrugiolo@ice.ufjf.br,
lorenza@ice.ufjf.br, ssoares@ice.ufjf.br

RESUMO

O problema de roteamento de veículo elétrico com janelas de tempo e recargas parciais é uma variação do VRPTW, onde é permitido aos veículos elétricos pararem em estações de recarga para uma recarga parcial ou total da bateria. O objetivo é minimizar a quantidade de veículos necessária para atender os clientes e o tempo total gasto nas rotas dos veículos (isto é, tempos de viagem, tempo de carregamento e tempos de espera). Este trabalho propõe um GRASP Reativo, composto por um algoritmo construtivo, refinamentos de solução e uma busca local via RVND. Os resultados numéricos mostram que o método gera soluções com custos competitivos em relação à literatura e com tempo de processamento significativamente inferior.

PALAVRAS CHAVE. Roteamento de veículo elétrico, logística verde, GRASP reativo.

Tópicos (Metaheurística, Logística e Transporte)

ABSTRACT

The Electric Vehicle-Routing Problem with Time Windows and Partial Recharges is a VRPTW variant in which eletric vehicles are allowed to stop in recharging stations to a fully or partial recharge. The goal is to minimize the amount of vehicles required to attend the customers and the total time spent in the vehicle's routes (i.e., travel times, charging times and waiting times). This work proposes a Reactive GRASP, composed by a constructive algorithm, solution refinements and a local search. Numerical results show that the method generates solutions relatively close, and sometimes better than the ones in the literature, spending significantly less processing time.

KEYWORDS. Electric Vehicle Routing Problem, Green Logistic, Reactive GRASP.

Paper topics (Metaheuristic, Logistics and Transport)

Galoá { Este trabalho foi publicado utilizando o **Galoá** proceedings

1. INTRODUÇÃO

A preservação do meio ambiente, preocupação crescente em todo o mundo, tem motivado mudanças significativas em vários setores da sociedade. Muitos governos têm investido em fontes de energia limpa e ampliado as regulamentações sobre emissões de poluentes e exploração de recursos não renováveis. Nas organizações do setor produtivo, o termo "logística verde" vem sendo usado para identificar processos logísticos que apresentam um equilíbrio entre a eficiência econômica e os impactos ambientais, considerando a sustentabilidade e o uso racional de recursos (Sbihi e Eglese [2010]).

Entre as diversas iniciativas governamentais com significativo impacto na área de logística verde, destaca-se a decisão de alguns países de banir o uso de veículos à combustão a partir de 2030 (Dekker et al. [2012]). Em curto período de tempo é prevista a substituição destes veículos, com grande potencial de poluição atmosférica e sonora, por veículos elétricos, que podem ser recarregados através de energia renovável, não emitem gases poluentes e não causam poluição sonora. Tal mudança tem reflexos consideráveis no planejamento de transportes, uma vez que, em geral, veículos elétricos têm menor autonomia. O alto custo das baterias, a localização (ou escassez) de estações de recarga e o tempo expressivo necessário para esta operação são fatores que limitam as possibilidades de roteamento de frotas. Assim, devido à baixa autonomia, o planejamento da rota de veículos elétricos deve considerar recargas ao longo do trajeto, prevendo o tempo desta operação e evitando grandes desvios, especialmente se o número de estações de recarga for escasso.

No meio acadêmico, boa parte dos problemas da logística de transporte são modelados através do problema de roteamento de veículos (VRP) e suas variações. No VRP, uma frota de veículos deve ser utilizada para atender demandas de clientes, a custo mínimo, de modo que cada cliente seja visitado exatamente uma vez e a capacidade dos veículos seja respeitada. Variações do problema consideram, por exemplo, janelas de tempo para atendimento dos clientes (VRPTW), frota heterogênea (HFVRP) ou possibilidade de entregas fracionadas (SDVRP). Com o advento dos veículos elétricos, surgiram variantes do VRP que consideram a autonomia do veículo e abordam características específicas, como a existência de estações de troca de bateria ou de estações de recarga, com obrigatoriedade de recarga total ou a possibilidade de recarga parcial (Schneider et al. [2014], Bruglieri et al. [2017]).

O problema de roteamento elétrico com janelas de tempo e recarga parcial (E-VRPTW-PR), proposto por Bruglieri et al. [2017], consiste em determinar rotas de veículos elétricos para atendimento das demandas dos clientes, sendo que:

- toda a demanda do cliente precisa ser atendida, de uma só vez, cumprindo o tempo de serviço definido para cada cliente;
- o cliente deve ser visitado no período estipulado pela sua janela de tempo, sendo permitido ao veículo ficar parado, se necessário for, para aguardar o início do período de atendimento;
- a rota de cada veículo começa e termina no depósito e a soma das demandas dos clientes da rota não pode ultrapassar a capacidade de carga do veículo;
- a autonomia do veículo deve ser considerada e, em caso de necessidade, a bateria do veículo pode ser recarregada total ou parcialmente nas estações de recarga disponíveis.

O objetivo principal é minimizar a quantidade de veículos utilizados no atendimento, bem como o tempo total gasto pelos veículos em suas rotas (tempo de deslocamento, mais tempo de recarga e de espera).

O E-VRPTW-PR pode ser modelado sobre um grafo direcionado $G = (V, A)$, ponderado nos arcos, onde $V = D \cup F$ é o conjunto de vértices, sendo D o conjunto de clientes e F o conjunto de estações de recarga e depósito. Para cada arco $(i, j) \in A$, são conhecidos a distância d_{ij} e o tempo de viagem t_{ij} entre os vértices i e j. Para cada cliente $i \in D$, são conhecidos a demanda $q_i \geq 0$, a janela de tempo $[e_i, l_i]$ em que a entrega pode ser realizada, e o tempo de serviço $s_i \geq 0$ necessário para a efetivação da entrega. São conhecidos ainda o intervalo $[e_0, l_0]$, que especifica o período de tempo em que os veículos podem sair e entrar no depósito, que também pode ser utilizado

como estação de recarga ao longo das rotas dos veículos. Quanto aos veículos, considera-se frota homogênea, e s ão conhecidos, a capacidade de carga de transporte C, a capacidade da bateria Q, o consumo de bateria r e a taxa de recarga g. Em qualquer estação de recarga, o tempo de recarga ρ é linearmente proporcional a g. O tempo total da rota de um veículo compreende o período entre o instante em que este deixa o depósito pela primeira vez até o instante em que retorna ao depósito pela última vez. No problema proposto em Bruglieri et al. [2017], objeto de pesquisa abordado nesse trabalho, o objetivo é, prioritariamente, minimizar a quantidade de veículos utilizados na solução e, então, o tempo total gasto em cada rota.

A Figura 1 ilustra uma solução ótima para uma instância do problema. No depósito D, são indicados a janela de tempo (JT), bem como o instante de saída (TS) e de chegada (TC) de cada rota. Para cada cliente, são informados a janela de tempo (JT), a bateria remanescente (BR) e o instante de chegada (TC). A cada parada em uma estação de recarga, são informados o tempo em que o veículo permanece parado para o abastecimento (TR) e a bateria remanescente no veículo após a recarga (BR). Neste exemplo simplificado, o peso de cada arco indica, simultaneamente, a distância entre os pontos, o tempo de viagem e a quantidade de bateria gasta para percorrê-lo. O tempo de serviço em cada cliente é fixo em 10 unidades e a capacidade da bateria nos veículos é 77. Em todas as estações, a bateria é recarregada a uma taxa de 2,5 unidades por instante de tempo. Considere que a capacidade do veículo é superior à soma de todas as demandas.

No exemplo, operações de recarga são necessárias em ambas as rotas, sendo que, na rota 2, duas paradas são realizadas na estação $E1$. Observe que, se a rota 2 tivesse uma única operação de recarga na estação $E1$ após visitar o cliente $C5$, a autonomia do veículo seria respeitada, mas não seria possível alcançar o cliente $C3$ antes do término de sua janela de tempo. Por isso, a bateria é totalmente recarregada antes da visita ao cliente $C5$, permitindo que o restante da bateria necessário para completar toda a rota seja recarregado rapidamente entre os clientes $C5$ e $C3$.

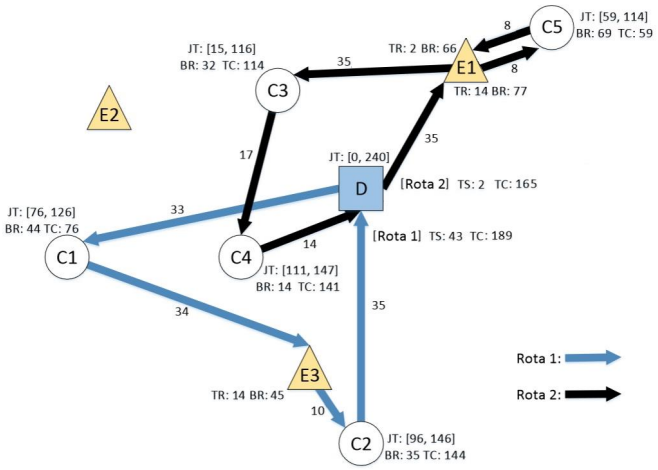


Figura 1: Exemplo de solução

O restante do texto está assim organizado: na Seção 2, é realizada uma revisão da literatura; a Seção 3 descreve o algoritmo proposto para o E-VRPTW-PR. Os resultados computacionais são apresentados na Seção 4 e, por fim, as conclusões e propostas de trabalhos futuros são discutidas na Seção 5.

2. TRABALHOS RELACIONADOS

Assim como o E-VRP-TW, o problema de otimização E-VRPTW-PR é uma variação do VRPTW, cuja complexidade é NP-difícil. Assim, não se conhece algoritmo que resolva o problema na otimalidade em tempo polinomial, o que desencoraja o uso de abordagens exatas para resolver

instâncias de tamanho real. O uso de meta-heurísticas puras ou combinadas com métodos exatos é comumente encontrado em trabalhos da literatura.

O E-VRPTW, introduzido em Schneider et al. [2014], lida com a possibilidade de recarga em qualquer estação disponível, sendo que a operação de recarga sempre completa toda a carga da bateria e, por isso, o tempo de recarga varia com o nível da bateria do veículo ao chegar na estação. Nesse trabalho, dois objetivos hierárquicos foram minimizados, primeiro, o número de veículos presentes na solução e, depois, a distância total percorrida nas rotas. O problema foi solucionado usando algoritmo exato sobre um modelo de programação linear inteira mista, combinado com uma meta-heurística híbrida.

O E-VRPTW-PR foi tratado em Keskin e Çatay [2016], com o objetivo de minimizar a distância total de viagem dos veículos. A abordagem proposta integra quatro heurísticas: remoção de cliente, inserção de cliente, remoção de estação e inserção de estação. Em Hiermann et al. [2016], os autores utilizam uma frota heterogênea de veículos elétricos, cada um com um custo associado, minimizando os custos de aquisição e a distância total percorrida. O problema é resolvido pelo uso de um algoritmo de branch-and-price e o trabalho ainda propõe uma heurística híbrida, que combina uma busca adaptativa na vizinhança com busca local e procedimento de rotulação.

Uma variação do problema de roteamento de veículos elétricos que considera, além de frota heterogênea, o tempo de recarga e um tempo de viagem variável do cliente foi tratada em Shao et al. [2017]. Neste trabalho, os autores propõem uma abordagem evolucionista que retorna o conjunto de rotas indicando o horário de saída do depósito para cada veículo e o plano de recarga dos mesmos. Além do algoritmo genético, utiliza ainda um algoritmo de caminho mínimo baseado no algoritmo de Dijkstra para obter caminhos mais curtos entre os vértices ao longo das rotas.

Uma abordagem híbrida para o E-VRPTW-PR, composta por três fases, foi proposta em Bruglieri et al. [2017]. Na primeira fase, os autores resolvem uma versão simplificada do problema utilizando o modelo matemático de Erdoğan e Miller-Hooks [2012] para o GVRP, com janelas de tempo e restrições de capacidade. Na fase seguinte, a sequência dos vértices das rotas encontradas na fase 1 é fixada no modelo matemático para o E-VRPTW-PR (proposto pelos autores em Bruglieri et al. [2015]). Neste caso, o modelo é utilizado para otimizar o tempo de uso dos veículos, alterando os instantes de chegada e partida em cada vértice. Na última fase, com o mesmo modelo matemático, a solução obtida na fase 2 é o ponto de partida para o uso combinado de VNS - *Variable Neighborhood Search* e *Local Branching*.

3. ALGORITMOS PROPOSTOS

A metaheurística GRASP, *Greedy Randomized Adaptive Search Procedure*, proposta por Feo e Resende [1995], é um algoritmo iterativo em que cada laço envolve duas fases: construção e busca local. Na fase de construção é gerada uma solução através de um algoritmo guloso, randomizado e adaptativo. Em seguida, a solução é submetida a um procedimento de busca local. A abordagem proposta neste trabalho para o E-VRPTW-PR consiste em um algoritmo GRASP Reativo [Prais e Ribeiro, 2000], onde o parâmetro de aleatoriedade do algoritmo construtivo é ajustado durante a execução do GRASP, combinado com um algoritmo RVND (*Random Variable Neighborhood Descent* [Subramanian et al., 2010]) que combina diferentes vizinhanças na busca local. Os detalhes da estratégia proposta são apresentados nas próximas subseções.

Uma vez que o objetivo principal do problema é minimizar o número de veículos, o algoritmo proposto considera este aspecto ao longo da busca pela melhor solução. Como esse fator não é considerado na função objetivo apresentada por Bruglieri et al. [2017], neste trabalho foi utilizada uma função auxiliar para guiar o processo de busca onde foi acrescentada uma segunda parcela incorporado um custo M a cada um dos k veículos utilizados. Usando esta função é possível comparar soluções com número diferente de veículos.

3.1. Algoritmo Construtivo VIP

Dadas as restrições do E-VRPTW-PR, obter uma solução viável por heurísticas construtivas é uma tarefa árdua. A estratégia proposta é baseada na construção paralela das rotas, onde as

rotas são definidas simultaneamente.

Para cada par (c, r), onde c é um cliente e r é uma rota, é calculado um custo associado a inserção do cliente c na rota r. A partir destes custos, a cada iteração do algoritmo, são selecionados o cliente (c) e a rota (r) em que este será inserido.

Durante o processo de construção, além do custo associado a cada par (c, r), considera-se também a urgência no atendimento ao cliente, de forma a dar maior prioridade àqueles clientes com menor folga em relação aos demais. Para indicar esta prioridade, foi definido o conceito de *Cliente VIP*. Para favorecer a viabilidade no que se refere à janela de tempo de atendimento, o conceito de Cliente VIP visa priorizar os clientes ainda não inseridos na solução, cujas janelas de tempo de atendimento estejam mais restritivas. Para tanto, a cada iteração do algoritmo construtivo, são considerados Clientes VIP em relação a uma rota j todos aqueles que possuem suas janelas de tempo relativas menores do que $1/m$ da média do tamanho das janelas relativas de todos os clientes, onde m é um inteiro escolhido pelo algoritmo. O conceito janela relativa de um cliente equivale ao tamanho da janela, caso o tempo de saída do último nó visitado na rota j, dito *tempo atual da rota*, seja menor que o tempo de abertura da janela deste cliente, ou a diferença entre o tempo de fechamento da sua janela e o tempo de saída do último nó, caso o tempo atual da rota seja maior que a abertura da janela. Note que, quanto maior o valor de m, menor será a quantidade de clientes VIP, visto que menor será o tamanho da janela necessária para o cliente ser classificado como tal. Vale observar que, durante a construção, a inserção de clientes ocorre sempre no final das rotas.

O construtivo proposto é apresentado no Algoritmo 1, onde o parâmetro α indica o grau de aleatoriedade do algoritmo e t indica o percentual das possíveis inserções a serem consideradas a cada operação de inserção.

Algoritmo 1 Construtivo VIP (α, t)

```
1:  procedure CONSTRUTIVO VIP
2:      m ← rand() mod 5 + 2
3:      qtd ← SomaDemandasClientes/capacidadeVeiculo
4:      S ← inicializa_Solucao(qtd)
5:      for cada veiculo j do
6:          inicializa(listaVIP_j, m)
7:          for cada cliente C ∈ listaVIP_j do
8:              adiciona a LI_j aqueles que podem ser inseridos antes de C e o próprio C
9:          Ordena LI_j (C1, C2, C3)
10:     while qtdClientesVisitados < numeroClientes do
11:         LMI ← melhoresInsercoes(t)
12:         if LMI vazio then
13:             Adiciona novo veículo
14:             LMI ← melhoresInsercoes(t)
15:         LMI ← ordena(LMI, C1, C2)
16:         (c, r) ← sorteio(LMI, α)
17:         S ← Insere cliente c na rota r
18:         if c ∈ listaVIP_r then
19:             for Cada veiculo s diferente de r do
20:                 Atualiza listaVIP_s e LI_s e ordena LI_s
21:         Atualiza ListaVIP_r e LI_r e ordena LI_r
```

Neste algoritmo, inicialmente é definido o valor de m, utilizado para indicar o status VIP dos clientes. Em seguida, na linha 3, é calculado o número mínimo de veículos necessário para atender as demandas dos clientes, *qtd*. Definido o valor de *qtd*, na linha 4 é criada uma solução S

com uma rota para cada um dos veículos. Neste passo, cada rota é inicializada com o depósito.

Em seguida, para cada veículo j é criada sua lista de inserções LI_j contendo os clientes candidatos a inserção na rota j (linhas 5 a 9). Para tal, na linha 6 é criada a $listaVIP_j$ contendo todos os clientes VIPs em relação ao veículo j. Logo após definir os clientes VIPs é definida a Lista de Inserções para o veículo j (LI_j). Em LI_j são inseridos todos os clientes VIPs, bem como os demais clientes que podem ser atendidos antes de um cliente VIP, mantendo a garantia de que o cliente VIP poderá ser atendido na sequência. Se a lista VIP de j estiver vazia, a Lista de Inserções LI_j será composta por todos aqueles clientes que podem ser inseridos na rota do veículo j.

Após construída, LI_j é ordenada (linha 9), considerando os critérios: tamanho da janela relativa (C1), tempo de espera (C2) e custo final de inserção, que inclui tempo de viagem, tempo de recarga (se houver) e tempo de espera (C3). Para ordenação é determinado um custo total, calculado pela combinação linear deste valores: κ * *C1* + ρ * *C2* + ϕ * *C3*.

Depois, enquanto houver cliente a ser atendido (linha 10), é criada uma lista de melhores inserções LMI (linha 11), contendo as *t* melhores inserções para cada veículo, obtidas a partir das LIs. Em LMI são incluídos pares (c,r) indicando o cliente c e a rota r em que deve ser inserido. Caso a lista LMI esteja vazia (linha 12), é detectada a necessidade de incluir um novo veículo na solução, dado que os clientes ainda não atendidos não podem ser inseridos nas rotas dos veículos atualmente presentes na solução. A rota do novo veículo é inicializada da mesma maneira que as demais (linha 13) e a LMI é atualizada (linha 14). Na linha 15, *LMI* é ordenada de acordo com o tempo de espera e o custo final de inserção. O par (c, r) é escolhido aleatoriamente entre os α porcento primeiros pares de LMI (linha 16).

A inserção do cliente é feita sempre no final da rota, e pode ser de dois tipos: Inserção Tipo 1 - apenas o cliente é inserido, caso a bateria seja suficiente para atender o cliente e chegar à estação de recarga mais próxima ou ao depósito; ou Inserção Tipo 2 - quando a bateria não é suficiente, é inserida uma estação de recarga, e então o cliente. Nesta etapa da abordagem proposta, mesmo o problema permitindo recarga parcial, sempre que uma estação de recarga é inserida na solução, é dado tempo suficiente para recarga total da bateria.

Em seguida, nas linhas 18 a 21, as listas VIP e listas de inserções dos veículos são atualizadas, e as LI são ordenadas.

Uma vez gerada uma solução, antes de submetê-la à fase de busca local, três estratégias de refinamento são aplicadas, sendo elas:

- R1 - Atrasar saída do depósito: Consiste em tentar atrasar a saída do veículo do depósito, reduzindo o tempo de espera em alguns clientes. Para isso, para cada cliente na rota foi calculado qual o tempo mais tarde em que ele poderia ser visitado, sem perder a janela de nenhum outro cliente no restante da rota.
- R2 - Redução última recarga: Consiste em tentar fazer com que a bateria do veículo, no retorno ao depósito, seja zero. Isto é feito diminuindo a última recarga feita por aquele veículo, recarregando somente o necessário para que ele complete a sua rota.
- R3 - Transformar tempo de espera em tempo de recarga: Dadas duas estações, onde uma destas possui alguma folga, busca-se a melhorar distribuição do tempo necessário de recarga a fim de transformar o tempo em espera que gerou a folga na estação em tempo em recarga.

Essas funções são executadas de forma que, se uma gera mudança na solução, as outras são executadas novamente, ou seja, o refinamento só acaba quando nenhuma das três funções consegue melhoria. Nesta etapa, a recarga parcial é considerada para refinamento da solução.

3.2. RVND

Variable Neighborhood Descend (VND) é uma estratégia de melhoramento local que é comumente usada como subordinada em um Variable Neighborhood Search [Hansen e Mladenović, 1999] ou outras meta-heurísticas. No VND, são empregadas diferentes estruturas de vizinhança N_1, ..., N_n, que são exploradas a partir de trocas sistemáticas. A solução final representa um ótimo local para todas as vizinhanças. Um bom desempenho do VND depende de uma boa escolha na ordem

em que serão aplicadas as estruturas de vizinhança. Tentando resolver esse problema, surgiu o RVND, que aplica, a cada chamada do método, as estruturas de vizinhança em uma ordem aleatória [Subramanian et al., 2010; Hu e Raidl, 2006].

Para explorar o espaço de soluções do E-VRPTW-PR, três diferentes movimentos foram utilizados. Cada movimento define uma vizinhança $N(.)$, e são descritos a seguir.

- N_1**(s) - Realocação de cliente em outra rota:** Consiste em retirar um cliente de uma rota e tentar inserí-lo em outra, no melhor ponto possível.
- N_2**(s) - Realocação de cliente na mesma rota:** Consiste em trocar a posição de um cliente dentro da mesma rota e ajustar as escolhas de recarga, a fim de gerar ganho no tempo da solução.
- N_3**(s) - Troca de Clientes Entre Rotas:** Dados dois clientes de rotas distintas, este movimento consiste em trocar a posição destes, ajustando as escolhas de recarga de bateria.

Estes três movimentos são ilustrados na Figura 2. Cada um destes movimentos, quando aplicado em relação a uma solução s, determina as soluções que pertencem à vizinhança de s, representada por N_i(s). Como estratégia de exploração da vizinhança, neste trabalho optou-se por utilizar o melhor aprimorante, ou seja, após explorar toda a vizinhança de s, é retornada a melhor solução s' encontrada, caso ela seja melhor que a solução de entrada s.

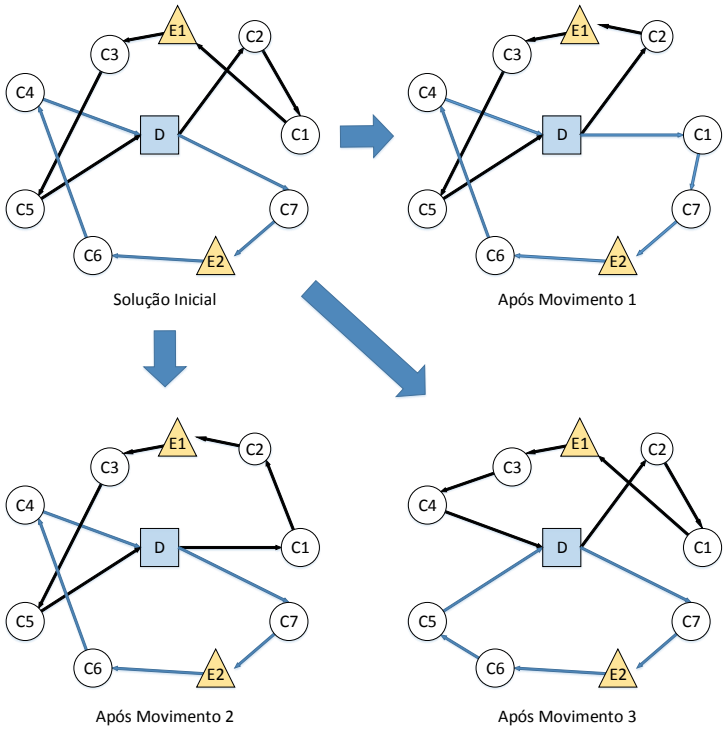


Figura 2: Movimentos de busca local

No processo de determinação das soluções que compõem N_i(s), após cada movimento realizado é verificada a viabilidade das rotas no que se refere às restrições relacionadas à janela de tempo e limite de carga da bateria. Para tratar uma eventual inviabilidade, a rota é percorrida até que seja identificado um ponto em que alguma das restrições não é atendida. Então, tenta-se viabilizar o trecho do começo até o ponto em questão. Neste cenário, dois tipos ajustes podem ser feitos:

- Realocação de Estações de Recarga: Retira-se uma estação intermediária anterior ao ponto de inviabilidade, na tentativa de realocá-la em outro ponto. Neste processo, entre as possíveis formas de sanar a inviabilidade, opta-se pela de menor custo.
- Inserção de Nova Estação de Recarga: Tenta-se inserir uma nova estação de recarga em todos

os pontos anteriores à inviabilidade, escolhendo a inserção que gera o menor acréscimo de custo na solução.

Esses ajustes de viabilização são executados em sequência, caso o primeiro não torne o trecho viável, o segundo é executado; com exceção do caso em que não existam estações de recarga no trecho do depósito até o ponto de inviabilidade, onde a única alternativa é inserir uma nova estação. Caso não seja possível a viabilização, o movimento que gerou a rota em questão é considerado inviável e a solução é descartada.

Após a viabilização das rotas, um processo simples de otimização é executado para verificar a possibilidade de ajustes na solução para reduzir o tempo total das rotas. Neste processo, uma rota é dividida em trechos compostos por 3 estações de recargas consecutivas (início, meio e fim) e os clientes entre elas. Cada trecho é submetido a uma função que verifica a possibilidade de reorganização das estações, realocando-as ou removendo-as, se possível, a fim de melhorar a qualidade da solução.

Como estratégia de combinação das três diferentes vizinhanças adotadas, optou-se pelo RVND, como apresentado no Algoritmo 2. A solução resultante deste algoritmo é um ótimo local em relação às três vizinhanças utilizadas.

Algoritmo 2 RVND

```
1:  procedure RVND(S*)
2:      input: r vizinhança em ordem randômica: N_1, N_2 e N_3.
3:      input: solução inicial s
4:      k ← 1
5:      while k ≤ r do
6:          encontrar a melhor solução s' de N_k(s)
7:          if s' melhor que s then
8:              s ← s'
9:              k ← 1
10:         else
11:             k ← k + 1
12:     output: s
```

3.3. GRASP Reativo

A metaheurística GRASP se destaca por ser uma estratégia simples de se implementar, eficiente para diversos tipos de problemas [Resende e Ribeiro, 2010] e por possuir apenas um parâmetro além do critério de parada: o parâmetro α responsável por ajustar o grau de aleatoriedade do algoritmo de construção. De acordo com estes mesmos autores, Resende e Ribeiro [2010], usar um único valor para o parâmetro α pode dificultar o processo de busca, além do mais, diferentes valores para α podem ser adequados para diferentes instâncias. Nesta direção, foi proposto o GRASP Reativo, uma extensão do GRASP onde o valor de α é ajustado dinamicamente durante a execução do algoritmo de acordo com a qualidade das soluções obtidas durante o processo de busca.

No Algoritmo 3, é possível verificar as etapas da estratégia proposta. Observe que, como parâmetros, tem-se o conjunto com m possíveis valores para α e *maxIter* e *maxISM* associados ao critério de parada do algoritmo. Enquanto *maxIter* corresponde ao número máximo de iterações, *maxISM* indica o número máximo de iterações sem melhora permitido pelo algoritmo.

Na linha 3, o vetor P é inicializado com a mesma probabilidade $1/m$, associada a cada valor candidato ao parâmetro α. A cada iteração do GRASP (linhas 6-17), o primeiro passo é determinar o valor de α a ser utilizado na iteração corrente, a chance de escolha de cada valor é indicada em P. Definido o valor de α, na linha seguinte uma solução é construída. Os três refinamentos apresentados na Seção 3.1 são aplicados na linha 9. Em seguida, a solução obtida é

Algoritmo 3 GRASP Reativo

```
1:  procedure GRASP REATIVO
2:      input: m possíveis valores para α, maxIter, maxISM.
3:      inicializaProbabilidades(P, m)
4:      S* ← ∅
5:      itSemMelhora ← 0
6:      for i ← 1...maxIter ou itSemMelhora < maxISM do
7:          α ← sorteio(P)
8:          S ← Construtivo VIP(α)
9:          S' ← Refinamentos(S)
10:         Sf ← RVND(S')
11:         if custo(Sf) < custo(S*) then
12:             S* ← Sf
13:             itSemMelhora ← 0
14:         else
15:             itSemMelhora++
16:         if i mod tBloco = 0 then
17:             ajustaProbabilidades (P)
```

submetida ao RVND. Caso a solução retornada pela estratégia de busca local seja melhor que S^*, esta é atualizada na linha 12 e a variável que contabiliza iterações sem melhora é zerada na linha 13. Caso contrário, incrementa-se a variável *itSemMelhora* na linha 15. A cada conjunto de *tBloco* iterações, as probabilidades de escolha dos valores de α são ajustadas considerando-se a média das soluções obtidas por cada α (linha 17), esta atualização é explicada em [Resende e Ribeiro, 2010].

4. RESULTADOS COMPUTACIONAIS

Para avaliar a eficiência dos algoritmos propostos, testes computacionais foram realizados sobre o conjunto de 36 instâncias utilizadas em Bruglieri et al. [2017] para o EVRPTW-PR. Na definição das instâncias, o número de clientes foi fixado em grupos com 5, 10 e 15 clientes, cuja distribuição geográfica deu-se de forma randômica, baseada em clusters, ou combinando estas duas formas, gerando os grupos C, R e RC, respectivamente. Dentro de cada grupo, diferentes instâncias foram criadas conforme combinações de valores de capacidade de bateria, capacidade de carga e taxa de recarga de bateria g. Para maiores detalhes sobre as instâncias, veja Schneider et al. [2014].

Os algoritmos foram desenvolvidos na linguagem C++ e executados em um PC AMD FX 6300 de 3.5Ghz, com 8 GB de memória, executando Linux Ubuntu 18.04.3 LTS. Cada instância foi submetida a 30 execuções independentes e o melhor resultado foi comparado com aqueles obtidos pelas abordagens híbridas propostas em Bruglieri et al. [2017], onde os autores utilizaram como ambiente de testes um PC Intel Core i7, 3.20 GHz, 6GB RAM.

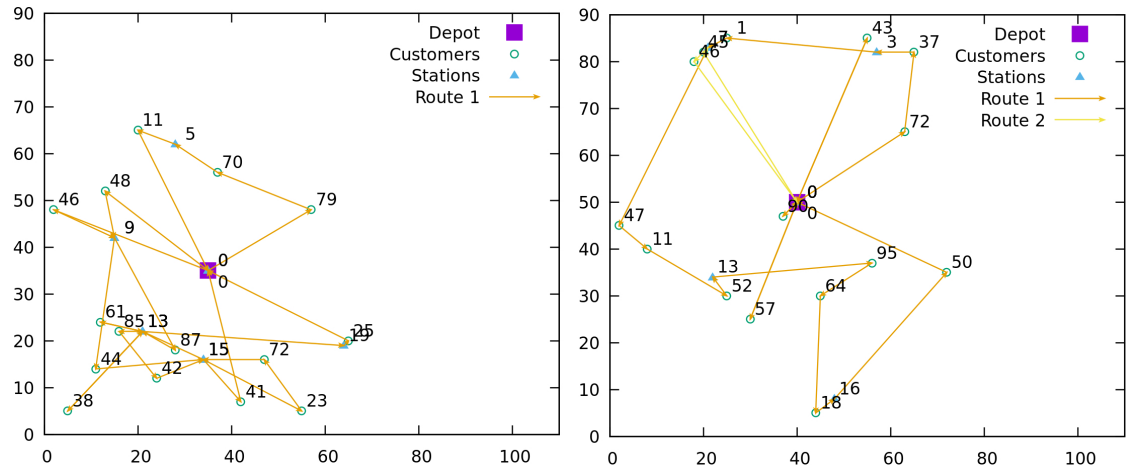
O conjunto de parâmetros do algoritmo GRASP proposto foi estabelecido a partir de testes empíricos com diferentes cenários, tendo-se chegado à seguinte configuração: critério de parada: 1.000 iterações ou 70 iterações sem melhora; tamanho do bloco de atualização do vetor de probabilidades: 50 iterações; no Algoritmo 1, fixou-se *t* em 80%, $\kappa = 0.25$, $\rho = 0.5$ e $\phi = 0.25$; α, usado como parâmetro para determinar a porcentagem em que será realizado o sorteio das inserções, tem valores entre 0.3, a 0.9. Após uma análise inicial das características das instâncias e dos custos das soluções, para função guia adotou-se $M = 1000$.

A Tabela 1 apresenta a comparação entre o GRASP proposto e os algoritmos VNSB e TPM de Bruglieri et al. [2017]. Na coluna Instância, os números ao final do nome indicam a quantidade de clientes. Para cada algoritmo, são mostrados o *Custo* (somatório do tempo de deslocamento, tempo de atendimento, tempo de espera e tempo de recarga); *nV*, número de veículos

utilizados; e *T(s)*, tempo de processamento em segundos. Os valores em negrito indicam os melhores resultados e a coluna *Desvio* indica o *gap* entre o algoritmo proposto e a melhor solução da literatura.

Observa-se que, sobre o primeiro objetivo do problema (número de veículos), a abordagem proposta obteve os mesmos resultados da literatura em 35 das 36 instâncias testadas. Conforme apresentado em Bruglieri et al. [2017], para as instâncias com cinco clientes, os valores indicam a solução ótima conhecida. Para este grupo de 12 instâncias, o GRASP obteve soluções com custos iguais aos obtidos pelos algoritmos da literatura em nove delas e, onde não alcançou estes resultados, não houve diferença significativa nos custos das soluções. O tempo médio da abordagem proposta ficou em torno de 61 vezes inferior ao algoritmo de três fases TPM, que apresentou os melhores resultados em Bruglieri et al. [2017]. O GRASP não alcançou os melhores resultados em somente três das 12 instâncias com 10 clientes, porém com um *gap* médio de apenas $0,20\%$ e com tempo quase 16 vezes inferior ao algoritmo mais eficiente. Para o grupo de 12 instâncias com 15 clientes, a abordagem proposta obteve soluções melhores em sete delas e o *gap* médio neste conjunto de instâncias foi -5.72%, indicando melhora em relação à literatura, e com tempo mais que 14 vezes inferior ao tempo obtido no trabalho de referência, onde os autores utilizam um hardware superior ao utilizado nos experimentos realizados.

Vale ressaltar que, para a instância R202-15, o custo obtido é maior do que o apresentado na literatura, mas como o número de veículos utilizados na solução é menor e este critério é hierarquicamente superior ao tempo total das rotas, a solução obtida é melhor que a apresentada na literatura. A Figura 3(a) mostra a rota obtida para esta instância. Destaca-se ainda a solução obtida para a instância RC202-15, Figura 3(b), onde o algoritmo proposto alcançou solução com custo $48,39\%$ inferior ao melhor dos dois algoritmos da literatura.



(a) R202-15 (b) RC202-15

Figura 3: Exemplo de soluções para as instâncias R202-15 e RC202-15

5. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho traz uma proposta de algoritmo GRASP reativo para o E-VRPTW-PR em que, na fase de construção, utiliza-se o conceito de lista VIP para a inclusão de clientes nas rotas e, na fase de busca local, faz uso de um algoritmo do tipo RVND, onde são empregadas três estruturas de vizinhança e dois procedimentos de melhoria em termos de viabilização e de otimização de rotas. Testes computacionais mostraram que, com tempo de processamento muito inferior às abordagens da literatura, foi possível obter soluções com custos melhores ou iguais em 33 das 36 instâncias testadas, sendo que em sete delas o algoritmo proposto obteve soluções estritamente melhores.

Embora não existam trabalhos relacionados em que sejam apresentados resultados para instâncias cujo número de clientes alcance valores mais realistas em termos de aplicação, os tempos

Tabela 1: Tabela de comparação com resultados da literatura

	VNSB			TPM			GRASP-RVND			
Instância	Custo	nV	T(s)	Custo	nV	T(s)	Custo	nV	Desvio	T(s)
C101-5	**1.262,84**	2	0,70	**1.262,84**	2	0,81	**1.262,84**	2	0,00%	0,13
C103-5	**987,87**	1	0,29	**987,87**	1	0,28	**987,87**	1	0,00%	0,21
C206-5	**1.296,82**	1	14,16	**1.296,82**	1	0,56	1.296,85	1	0,00%	0,27
C208-5	**984,80**	1	1,31	**984,80**	1	1,42	984,82	1	0,00%	0,22
R104-5	**196,17**	2	2,43	**196,17**	2	0,07	**196,17**	2	0,00%	0,10
R105-5	**231,59**	2	0,33	**231,59**	2	0,09	**231,59**	2	0,00%	0,06
R202-5	**234,16**	1	1,62	**234,16**	1	0,34	234,17	1	0,00%	0,41
R203-5	**287,09**	1	1,81	**287,09**	1	0,31	**287,09**	1	0,00%	0,55
RC105-5	**314,31**	2	94,01	**314,31**	2	215,55	**314,31**	2	0,00%	0,09
RC108-5	**342,32**	2	1,75	**342,32**	2	0,39	**342,32**	2	0,00%	0,15
RC204-5	**264,86**	1	49,16	**264,86**	1	0,44	**264,86**	1	0,00%	0,55
RC208-5	**253,17**	1	8,00	**253,17**	1	0,42	**253,17**	1	0,00%	0,86
MÉDIAS			**14,63**			**18,39**			**0,00%**	**0,30**
C101-10	**2.335,20**	3	183,43	**2.335,20**	3	545,76	**2.335,20**	3	0,00%	1,17
C104-10	1.611,72	2	73,43	1.584,90	2	252,95	**1.584,89**	2	0,00%	3,91
C202-10	2.949,77	1	5,62	2.949,77	1	3,83	**2.949,74**	1	0,00%	3,21
C205-10	**2.525,77**	2	0,79	2.539,88	2	1,45	2.539,88	2	0,56%	2,62
R102-10	**443,62**	3	33,96	**443,62**	3	0,35	**443,62**	3	0,00%	1,00
R103-10	**347,70**	2	93,49	**347,70**	2	12,19	**347,70**	2	0,00%	1,83
R201-10	536,38	1	21,70	**536,05**	1	76,03	545,69	1	1,80%	5,98
R203-10	**527,68**	1	4,62	**527,68**	1	9,1	**527,68**	1	0,00%	10,53
RC102-10	**571,26**	4	59,01	**571,26**	4	24,76	**571,26**	4	0,00%	0,67
RC108-10	**493,23**	3	143,71	**493,23**	3	12,73	**493,23**	3	0,00%	1,55
RC201-10	793,52	1	40,46	**793,33**	1	9,25	793,52	1	0,02%	5,02
RC205-10	**611,66**	2	37,51	**611,66**	2	4,70	**611,66**	2	0,00%	6,95
MÉDIAS			**58,14**			**79,43**			**0,20%**	**3,70**
C103-15	5.254,93	3	240,46	2.446,99	3	178,86	**2.352,85**	3	-3,85%	14,80
C106-15	2.173,09	3	106,30	2.173,09	3	213,12	**2.088,63**	3	-3,89%	4,27
C202-15	3.664,12	2	139,83	**3.599,80**	2	61,15	**3.599,80**	2	0,00%	25,06
C208-15	**2.819,47**	2	591,57	**2.819,47**	2	72,26	**2.819,47**	2	0,00%	22,97
R102-15	1.241,80	5	559,52	**656,37**	5	113,75	656,39	5	0,00%	3,53
R105-15	567,82	4	487,02	567,82	4	2,23	**567,81**	4	0,00%	4,45
R202-15	839,74	2	349,29	840,99	2	58,96	956,22	**1**	*	68,14
R209-15	621,79	1	330,38	640,64	1	5.718,31	**611,23**	1	-1,70%	69,15
RC103-15	1.074,90	4	251,55	614,13	4	31,81	**614,12**	4	0,00%	6,12
RC108-15	974,88	3	756,61	596,17	3	288,01	**573,68**	3	-3,77%	7,74
RC202-15	3.222,49	2	444,54	1.611,29	2	71,87	**831,53**	2	-48,39%	36,51
RC204-15	685,61	1	617,22	685,86	1	4.119,53	**676,76**	1	-1,29%	79,82
MÉDIAS			**406,19**			**910,82**			**-5,72%**	**28,55**

Galoá { Este trabalho foi publicado utilizando o **Galoá** proceedings

computacionais obtidos nos experimentos realizados sugerem que a abordagem proposta apresenta escalabilidade melhor que as abordagens da literatura.

Como trabalhos futuros, uma vez que o problema traz dois objetivos com hierarquia bem definida, propõe-se o estudo e desenvolvimento de estratégias de penalização destes dois objetivos ao longo da exploração do espaço de busca.

Referências

Bruglieri, M., Mancini, S., Pezzella, F., Pisacane, O., e Suraci, S. (2017). A three-phase matheuristic for the time-effective electric vehicle routing problem with partial recharges. In *ScienceDirect*, volume 58, p. 95–102. Elsevier.

Bruglieri, M., Pezzella, F., Pisacane, O., e Suraci, S. (2015). A variable neighborhood search branching for the electric vehicle routing problem with time windows. *Electronic Notes in Discrete Mathematics*, 47:221 – 228. ISSN 1571-0653.

Dekker, R., Bloemhof, J., e Mallidis, I. (2012). Operations research for green logistics – an overview of aspects, issues, contributions and challenges. *European Journal of Operational Research*, 219 (3):671 – 679. ISSN 0377-2217.

Erdoğan, S. e Miller-Hooks, E. (2012). A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114.

Feo, T. A. e Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, p. 109–134.

Hansen, P. e Mladenović, N. (1999). An introduction to variable neighborhood search. *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, p. 433–458.

Hiermann, G., Puchinger, J., Ropke, S., e Hartl, R. F. (2016). The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research*, 252(3):995–1018.

Hu, B. e Raidl, G. R. (2006). Variable neighborhood descent with selfadaptive neighborhood-ordering. In *Proc. 7th EU/MEeting on Adaptive, Self-Adaptive, and Multi-Level Metaheuristics*.

Keskin, M. e Çatay, B. (2016). Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 65:111–127.

Prais, M. e Ribeiro, C. C. (2000). Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment. *INFORMS Journal on Computing*, 12(3):164–176.

Resende, M. G. e Ribeiro, C. C. (2010). *Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications*, p. 283–319. Springer US, Boston, MA.

Sbihi, A. e Eglese, R. (2010). Combinatorial optimization and green logistics. In *Annals of Operations Research*.

Schneider, M., Stenger, A., e Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. In *Transportation Science*, volume 48, p. 500–520. Informs.

Shao, S., Wei, G., Ran, B., He, Z., e Bi, J. (2017). Electric vehicle routing problem with charging time and variable travel time. *Mathematical Problems in Engineering*, 2017:1–13.

Subramanian, A., Drummond, L., Bentes, C., Ochi, L., e Farias, R. (2010). A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers Operations Research*, 37(11):1899 – 1911. ISSN 0305-0548.