São José dos Campos - SP
LV SBPO
LV Simpósio Brasileiro de Pesquisa Operacional
São José dos Campos, SP - 6 a 9 de novembro de 2023

Um Algoritmo *Multi-Start Iterated Greedy* para o Problema de Roteamento de Veículos com Drones e Janela de Tempo

Marcelo Ian Rezende Menezes, Stênio Sã Rosário Furtado Soares,
Luciana Brugiolo Gonçalves, Lorenza Leão Oliveira Moreno
Universidade Federal de Juiz de Fora
Juiz de Fora, Minas Gerais, Brasil
{marcelo.ian, ssoares, lbrugiolo, lorenza}@ice.ufjf.br

RESUMO

O roteamento de veículos em áreas urbanas combinando caminhões e drones propicia vantagens no processo de entrega de mercadorias, como a redução no tempo total de atendimento, redução do custo de transporte e da emissão de CO_2, já que drones não fazem uso de combustíveis fósseis. Uma variação do problema de roteamento de veículos - VRP (*Vehicle Routing Problem*) onde se utiliza uma frota combinada de caminhões e drones e se considera janelas de tempo nos clientes, abreviada como VRP-DTW, é abordada neste trabalho. É proposto um algoritmo *Multi-Start Iterated Greedy* (MIG) combinado com *Random Variable Neighborhood Descend* (RVND). Os resultados preliminares são comparados com outras abordagens na literatura e demonstram que abordagem proposta é competitiva em termos de valor das soluções obtidas e em relação à eficiência de tempo computacional.

PALAVRAS CHAVE. Problema de Roteamento Verde, Roteamento de drones, IG, RVND.

Tópicos (L&T — Logística e Transportes, MH — Metaheuristicas, IC — Inteligência Computacional)

ABSTRACT

Routing vehicles in urban areas combining trucks and drones provides advantages in the process of delivering goods, such as reducing total service time, reducing transportation costs and CO_2 emissions, as drones do not use fossil fuels. A variation of the vehicle routing problem (VRP) that involves a fleet of trucks and drones, considering time windows for customers, abbreviated as VRP-DTW, is addressed in this study. A Multi-Start Iterated Greedy (MIG) algorithm combined with a Random Variable Neighborhood Descend (RVND) is proposed. Preliminary results are compared with other approaches in the literature, demonstrating that the proposed approach is competitive in terms of solution quality and computational time efficiency.

KEYWORDS. Green Routing Problem, Drone Routing, Iterated Greedy, RVND.

Paper topics (L&T — Logistics and Transport, MH — Metaheuristics, IC — Computational Intelligence)

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

1. Introdução

A busca por soluções eficientes no transporte de produtos tem sido um desafio constante em diversos setores da sociedade. Nesse contexto, a combinação estratégica de caminhões e drones surgiu como uma alternativa promissora para otimizar o roteamento de veículos ao agilizar as entregas na última etapa do processo de entregas de mercadorias, ou seja, o transporte de produtos do centro de distribuição até o cliente final.

No problema de roteamento de veículos - VRP *(Vehicle Routing Problem)*, há uma demanda crescente em aplicações reais por soluções que visem reduzir os custos operacionais, ao mesmo tempo em que otimizam o tempo de entrega, minimizam a emissão de gases poluentes e diminuem o tempo de espera dos clientes. Nesse prisma, conforme abordado por Wang e Sheu [2019], o Problema de Roteamento de Veículos com Drones (VRP-D) apresenta-se como um recurso promissor para agilizar o processo de transporte, considerando os vários aspectos relacionados à logística verde.

No entanto, em [Wang et al., 2017], o autor destaca as dificuldades de se implementar a cooperação entre caminhão e drone, seja por dificuldades tecnológicas ou por problemas legislativos e regulamentares. Porém, os benefícios não podem ser desconsiderados. Em [Chiang et al., 2019], evidencia-se a preocupação com a sustentabilidade ao levar em conta a emissão de gases poluentes ao longo das rotas dos veículos a combustão, e demonstra-se que o uso de drones contribui positivamente para a logística verde, uma vez que esses veículos poluem menos do que os caminhões.

O problema do caixeiro viajante (TSP) usando drones foi introduzido por Murray e Chu [2015], onde os autores sugerem duas variações: o *flying sidekick traveling salesman problem* (FSTSP) e o *parallel drone scheduling traveling salesman problem* (PDSTSP). Em [Wang et al., 2017], o problema é novamente abordado, desta vez considerando uma frota com múltiplos caminhões - VRPD *(Vehicle Routing Problem with Drones)*.

Neste trabalho aborda-se uma variação do VRP-D onde considera-se a janela de tempo nos clientes, referenciada na literatura como Problema de Roteamento de Veículos com Drones e Janela de Tempo, VRP-DTW (do inglês *Vehicle Routing Problem With Drones and Time Windows*). Em [Faria et al., 2022], os autores consideraram restrições de janelas de tempo dos clientes e o objetivo de minimização do custo monetário para fazer todas as entregas usando uma frota de caminhões e um conjunto de drones.

Devido à complexidade desses problemas, métodos exatos para solucioná-los são aplicáveis apenas para instâncias com um número pequeno de clientes, geralmente até 15. No entanto, considerando que problemas reais envolvem uma quantidade significativamente maior de clientes, as técnicas de computação inteligente têm sido amplamente exploradas, como evidenciado em Lei et al. [2022], onde o autor propõe uma abordagem baseada em Colônia Artificial de Abelhas para o problema onde busca-se minimizar simultaneamente a energia total consumida pelos caminhões, a energia total consumida pelos drones e o número total de caminhões.

Em Sacramento et al. [2019] é utilizado um algoritmo ALNS (*Adaptive Large Neighborhood Search*) para uma variação do problema onde não se considera janela de tempo. Euchi e Sadok [2021] apresenta um Algoritmo Genético híbrido para o VRPD, enquanto Schermer et al. [2018] propõem uma abordagem VNS (*Variable Neighborhood Search*) para o mesmo problema.

Di Puglia Pugliese et al. [2021] apresenta uma heurística de duas fases com *multi-start* para o VRP-DTW e em Faria [2022] é apresentado um algoritmo *Greedy Randomized Adaptive Search Procedure* - GRASP com RVND e uma fase de refinamento baseada em um Modelo de Programação Inteira para o mesmo problema.

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

Neste trabalho está sendo proposto um algoritmo *Multi-Start Iterated Greedy* (MIG) combinado com um *Random Variable Neighborhood Descend* (RVND) para tratar o VRP-DTW. O restante do artigo está assim estruturado: na Seção 2 há a definição formal do problema, na Seção 3 está o algoritmo proposto. Na Seção 4 estão informações acerca das instâncias e dos experimentos realizados e a Seção 5 traz as conclusões e sugestões de trabalhos futuros.

2. Definição do Problema

O VRP-DTW consiste numa variação do problema clássico de roteamento de veículos capacitados com janela de tempo. Nesse cenário, o objetivo é determinar rotas eficientes para múltiplos veículos, levando em consideração a capacidade de carga dos veículos e as janelas de tempo dos clientes a serem atendidos.

Conforme apresentado por Faria et al. [2022], o VRP-DTW pode ser modelado sobre os grafos ponderados nas arestas $G_c(V, A^c)$ e $G_d(V_d, A^d)$, onde $V = \{v_0, v_1, v_2, \ldots, v_n\}$ representa os nós, A^c é o conjunto de arestas (v_i, v_j) cujos pesos indicam a distância entre os vértices caso se utilize caminhão e A^d é o conjunto de arestas (v_i, v_j) cujos pesos indicam a distância quando se utiliza drone. Seja v_0 o depósito, $V^* = V \setminus \{v_0\}$ o conjunto de clientes a serem atendidos, $V_d \subset V^*$ o conjunto de clientes que, por característica de suas demandas, podem ser atendidos por drones, A^c_{ij} a distância entre v_i e v_j por caminhão e A^d_{ij} a distância entre v_i e v_j por um drone.

Cada cliente $v_i \in V^*$ deve ter sua demanda q_i completamente atendida por um único caminhão ou drone dentro de uma janela de tempo $[a_i, b_i]$, ou seja, o cliente não poderá ser atendido após o instante b_i e só poderá ser atendido a partir do instante a_i. Neste caso, o caminhão ou drone que chegar ao cliente v_i antes do instante a_i deverá esperar no cliente até a abertura da janela, sendo que no caso de atendimento por drone, este poderá esperar até no máximo T instantes de tempo.

Neste trabalho aborda-se a versão capacitada com frota homogênea do VRP-DTW, denotada como CVRP-DTW. Assim, considera-se uma frota $K = \{k_0, k_1, k_2, \ldots, k_m\}$ composta de caminhões com igual capacidade W^c e que operam com a mesma velocidade v^c. Assim, o tempo de viagem t^c_{ij} de um caminhão que sai do cliente v_i até o cliente v_j é dado por $t^c_{ij} = \frac{A^c_{ij}}{v^c}$, com A^c_{ij} denotando a distância entre os clientes v_i e v_j. Além disso, também é levado em conta o tempo s^c_i de atendimento do caminhão ao cliente v_i e o custo c^c de viagem por unidade de distância percorrida por caminhão.

No VRP-DTW cada caminhão dispões de uma quantidade indefinida de drones, que decolam e pousam em seus respectivos veículos. Esses drones podem ser utilizados para atender às demandas enquanto o caminhão percorre sua rota. Semelhante ao que foi definido para os caminhões, cada drone possui uma mesma capacidade W^d, mesma velocidade v^d e tempo de viagem do cliente v_i ao cliente v_j é calculado como $t^d_{ij} = \frac{A^d_{ij}}{v^d}$, com tempo de atendimento s^d_i e custo de viagem c^d por unidade de distância percorrida por drone. Como os drones possuem energia limitada, considera-se uma distância máxima de voo para cada entrega. Além disso, o tempo de recarga ou troca de bateria dos drones é desprezado.

Cada drone atende um único cliente por vez. Assim, uma rota de drone é sempre da forma $< v_i, v_w, v_j >$, onde v_w é o cliente atendido e os clientes v_i e v_j são clientes atendidos por caminhão ao qual o drone está associado. Além disso, é necessário que o instante de chegada do drone ao cliente v_j se dê antes do caminhão já ter partido e também que o instante previsto para a decolagem de v_i seja após a chegada do caminhão neste cliente. No caso em que o drone chega a v_j antes do caminhão, é permitido que o drone espere pelo caminhão, desde que seja respeitado o limite T de tempo de espera estabelecido.

O objetivo do VRP-DTW é encontrar um conjunto de rotas de caminhão e de drones que atendam todos os n clientes, respeitando a capacidade dos veículos e a janela de tempo especificada

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

para cada cliente, de forma a minimizar o custo total de transporte. A função de minimização do custo total da solução é data por:

$$min\ Z = c^c \times (\sum_{i \in V} \sum_{j \in V} \sum_{k \in K} A^c_{ij} \times x^k_{ij}) + c^d \times (\sum_{a \in V} \sum_{b \in V} \sum_{c \in V} \sum_{k \in K} (A^d_{ab} + A^d_{bc}) \times y^k_{abc})$$

onde a variável x^k_{ij} é uma variável binária que assume o valor 1 se a aresta (v_i, v_j) estiver incluída na rota do caminhão k, e 0 caso contrário. De maneira similar, a variável binária y^k_{abc} assume o valor 1 se as arestas (v_a, v_b) e (v_b, v_c) estão sendo utilizadas na rota de um drone associado ao caminhão k, e 0 caso contrário. Desta forma, a primeira parcela se refere ao custo com as entregas feitas por caminhões, enquanto a segunda parcela computa o custo das rotas dos drones. O objetivo é minimizar o custo total.

3. Heurística Proposta

Nesta seção é apresentada a abordagem proposta para solucionar o Problema de Roteamento de Veículos Capacitados com Drones e Janelas de Tempo (CVRP-DTW). A abordagem combina um *Iterated Greedy* (IG) *Multi-Start* com um *Random Variable Neighborhood Descend* (RVND), onde a solução é construída a partir de um algoritmo guloso randomizado e depois submetido a várias etapas de destruição e reconstrução com refinamento.

3.1. Algoritmo Construtivo Guloso Randomizado

Para construir uma solução inicial foi desenvolvido um algoritmo guloso randomizado que consiste na inserção sistemática de clientes nas rotas de caminhão ou drones. O algoritmo proposto possui duas etapas, a primeira de atendimento aos clientes que são atendidos obrigatoriamente por caminhões, $V_c = V - V_d$, e a segunda onde os demais clientes que ainda não foram atendidos são inseridos na solução atendidos por drone ou por caminhão.

O Algoritmo 1 descreve o processo de construção de uma solução. Inicialmente os clientes são separados em dois grupos, $naoAtendidos$ e $naoAtendidosC$, que correspondem aos clientes dos conjuntos V_d e V_c, respectivamente. Na linha 3 a solução é inicializada, sendo criadas *minRoutes* rotas, cada uma com um par depósito-cliente, sendo os clientes escolhidos com base na sua demanda, os maiores sempre são escolhidos primeiro.

Nas linhas de 4 a 15 ocorre a primeira etapa da construção, onde são considerados apenas os clientes que, devido ao valor da demanda, obrigatoriamente devem ser atendidos por caminhão. A cada iteração é gerada a lista de candidatos *LC* que engloba todas as inserções possíveis de clientes em todas as posições de cada rota, gerando uma tupla *(cliente, rota, posição)*. Essa é uma lista ordenada por custo, ou seja, as inserções de menor custo aparecem primeiro nessa lista. O custo de ir de um cliente v_i a um cliente v_j é calculado por $cost_{ij} = A^c_{ij} \times c^c$.

Em seguida, se há inserções possíveis em LC, seleciona-se aleatoriamente uma inserção entre as $\alpha \times |LC|$ primeiras inserções de LC (linhas 6 e 7), com $\alpha \in]0, 1]$. A inserção é realizada na linha seguinte e o conjunto *naoAtendidosC* é atualizado removendo-se o cliente atendido. Se não houver inserção possível, nas linhas 12 e 13 cria-se uma nova rota, que é inicializada com o primeiro cliente do conjunto $naoAtendidosC$. O processo se repete até que $naoAtendidosC$ esteja vazio. A Figura 1(a) ilustra o fim desta primeira fase, onde os círculos verdes representam os clientes atendidos obrigatoriamente por caminhão e os azuis indicam os clientes que podem ser atendidos tanto por caminhão como por drones. Note na figura que, ao final desta etapa, o algoritmo criou duas rotas de caminhão sendo uma delas atendendo um único cliente (nó 25).

Na segunda etapa, linhas de 16 a 28, um processo semelhante é feito até que todos os demais clientes sejam atendidos. Nesta etapa a LC também inclui atendimentos feitos por drones, onde o custo é calculado incluindo a ida e a volta do drone.

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

Algorithm 1: Algoritmo Guloso

Input: $\alpha, minRoutes$
Output: *solução*

$naoAtendidos \leftarrow V^* - V_c$
$naoAtendidosC \leftarrow V_c$
$s \leftarrow inicializaRotas(minRoutes, naoAtendidos)$
while $naoAtendidosC \neq \emptyset$ **do**
  $LC \leftarrow addMovesPorCaminhao(s, naoAtendidosC)$
  if $LC \neq \emptyset$ **then**
    $candidato \leftarrow getRandom(LC, \alpha)$
    $s \leftarrow insereCliente(candidato)$
    $AtualizaLista(naoAtendidosC, candidato)$
  end
  else
    $s \leftarrow criaNovaRota(naoAtendidosC)$
    $AtualizaLista(naoAtendidosC)$
  end
end
while $naoAtendidos \neq \emptyset$ **do**
  $LC \leftarrow addMovesPorCaminhao(s, naoAtendidos)$
  $LC \leftarrow LC \cup addMovesPorDrone(s, naoAtendidos)$
  if $LC \neq \emptyset$ **then**
    $candidato \leftarrow getRandom(LC, \alpha)$
    $s \leftarrow insereCliente(candidato)$
    $AtualizaLista(naoAtendidos, candidato)$
  end
  else
    $s \leftarrow criaNovaRota(naoAtendidos)$
    $AtualizaLista(naoAtendidos)$
  end
end
return s

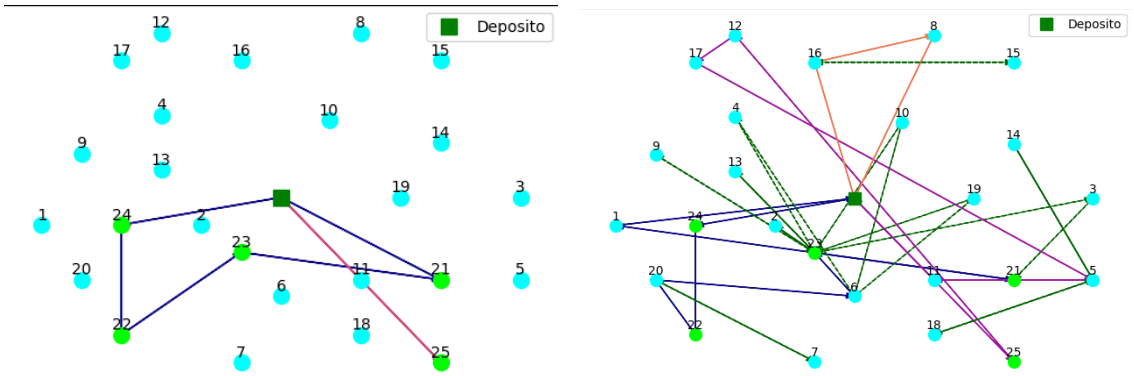
Em ambas as listas, só são consideradas inserções viáveis, ou seja, devem considerar a capacidade máxima do caminhão, as janelas de tempo e, no caso de drones, a inclusão só será possível caso o respectivo caminhão esteja disponível no nó de partida e, no caso da chegada, disponível dentro do tempo máximo de espera T.

Caso, durante o processo, a lista LC fique vazia e haja clientes restantes que não foram atendidos, uma nova rota de caminhão é criada para acomodar o primeiro cliente da lista de $naoAtendidos$ e o processo segue. A Figura 1(b) ilustra a solução final obtida pelo algoritmo, onde temos três rotas de caminhão, nas cores laranja (com dois clientes), rosa (com cinco clientes) e azul (com sete clientes), cada um com suas rotas de drone, tracejadas e na cor verde.

3.2. Destruição e Reconstrução

Uma solução gerada pelo algoritmo guloso passa por um processo de destruição e reconstrução, onde a proposta é remover $\beta\%$ dos clientes da solução e reinserí-los num processo

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br



(a) Solução parcial após a primeira etapa do algoritmo guloso

(b) Solução após o algoritmo guloso

Figura 1: Construção de uma solução em 2 etapas

semelhante ao algoritmo construtivo guloso. O Algoritmo 2 ilustra o processo.

Algorithm 2: Destroy e Reconstruct

Input: s, α, β
Output: *solução*
1 $nClientsToRemove \leftarrow numClientes(V) * \beta$;
2 $clientesRem \leftarrow \emptyset$
3 **while** $numClientes(clientesRem) < nClientsToRemove$ **do**
4 $\quad cr \leftarrow removeCliente(V^*, clientesRem)$
5 $\quad clientesRem \leftarrow clientesRem \cup cr$
6 $\quad$ **if** $atendidoPorCaminhao(cr)$ **then**
7 $\quad\quad clientesRem \leftarrow procuraERemoveDrones(cr)$
8 $\quad$ **end**
9 **end**
10 $s \leftarrow reconstroiCaminhos(clientesRem, \alpha)$
11 **return** s;

Inicialmente determina-se $nClientsToRemove$, o número mínimo de clientes a serem removidos. Os clientes removidos são armazenados numa lista semelhante à lista de não atendidos $naoAtendidos$ vista na Seção 3.1. Enquanto o número de clientes não removidos $clientesRem$ não alcançar $nClientsToRemove$, um cliente aleatório cr é selecionado na solução e removido. Se este cliente for atendido por caminhão (linha 6), deve-se verificar se existem clientes que são atendidos por drones que partem ou chegam em cr e, caso existam removê-los da solução também. Todos os clientes removidos são inseridos na lista $clientesRem$. Por fim, os clientes removidos são reinseridos na solução num processo semelhante ao que ocorre na segunda etapa do algoritmo guloso visto na Seção 3.1.

3.3. *Multi-start Iterated Greedy*

O procedimento detalhado na seção anterior é utilizado no Algoritmo 3. O parâmetro $maxIt$ define o número de iterações sem melhora, β indica o percentual máximo de clientes a seres removidos e $alpha$ define o percentual da lista de candidatos considerados no algoritmo guloso

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

randomizado. Observe que o valor de β varia ao longo do processo e é controlado pela variável auxiliar k. O objetivo é manter o parâmetro β em um determinado valor por algumas iterações.

Algorithm 3: Iterated Greedy

```
   Input: S, sBest, α, β, maxIt
   Output: sBest
 1 iter ← 0
 2 k ← 0
 3 while iter < maxIt do
 4 |   S' ← S
 5 |   S' ← DestroyAndReconstruct(S', α, β)
 6 |   S' ← RVND(S')
 7 |   k ← k + 1
 8 |   if custo(S') < custo(S) then
 9 |   |   if β > 0,1 & k > 25 then
10 |   |   |   β ← β − 0,05
11 |   |   |   k = 0
12 |   |   end
13 |   |   S = S'
14 |   |   if custo(S) < custo(SBest) & solucaoViavel(S) then
15 |   |   |   sBest ← S
16 |   |   end
17 |   end
18 |   else
19 |   |   iter ← iter + 1
20 |   |   if β < 0,8 & k > 25 then
21 |   |   |   β ← β + 0,05
22 |   |   |   k ← 0
23 |   |   end
24 |   end
25 end
26 return sBest;
```

A Figura 2 ilustra a solução que permanece com três rotas de caminhões, porém nota-se uma redução dos cruzamentos dos caminhões, principalmente na rota azul. Também há um pequeno aumento no número de clientes atendidos por drones, que agora fazem entregas mais curtas.

Conforme pode ser visto nas linhas 5 e 6, a solução passa por um processo de destruição e reconstrução, mostrado no Algoritmo 2 e depois submetido a um refinamento em um *Random Variable Neighborhood Descent* (RVND), que consiste numa adaptação do RVND proposto em Faria et al. [2022], com a inclusão de um movimento 2-opt clássico primeiro aprimorante viável.

Tendo em vista que no algoritmo guloso (Algoritmo 1) o número de rotas inicial é relevante na primeira parte do procedimento, conforme visto na Seção 3.1, neste trabalho está sendo proposto o uso do *Multi-start Iterated Greedy* (MIG) para possibilitar a chamada do IG (Algoritmo 3) com boas soluções e número variado de rotas iniciais. O Algoritmo 4 descreve o MIG.

Na linha 1, o número mínimo de rotas é definido como o menor número de caminhões

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

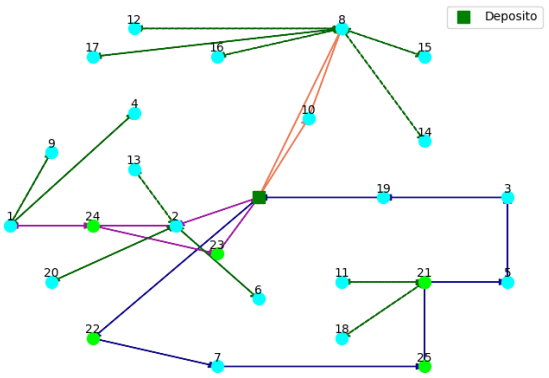


Figura 2: Solução após as iterações do IG

Algorithm 4: *Multi-start Iterated Greedy*

Input: $\alpha, \beta, nIter, maxIterIG$
Output: *sBest*

1 $minRotas \leftarrow demandaTotal(V)/capacidadeCaminhao$
2 $s \leftarrow AlgoritmoGuloso(0, minRotas)$
3 $s \leftarrow iteratedGreedy(s, SBest, \alpha, \beta, maxIterIG)$
4 $sBest \leftarrow s$
5 **for** $i \leftarrow 1$ **to** $nIter$ **do**
6 $\quad nRotas \leftarrow randomizaMinRotas(minRotas, nRotas)$
7 $\quad s \leftarrow AlgoritmoGuloso(\alpha, nRotas)$
8 $\quad sBest \leftarrow iteratedGreedy(s, SBest, \alpha, \beta, maxIterIG)$
9 **end**
10 **return** *sBest*;

necessários para atender todos os clientes, sendo calculado pela soma das demandas dos clientes dividido pela capacidade do caminhão. Na linha 2, uma solução inicial é gerada utilizando o algoritmo guloso usando valor 0 para o parâmetro α. Em seguida, na linha 3, a solução construída é dada como parâmetro de partida para o IG.

O número de iterações (*nIter*) indica o número de vezes que a solução será reconstruída. A cada iteração, o número de rotas que será passado como parâmetro no algoritmo guloso, *nRotas*, é randomizado na linha 6, podendo aumentar ou diminuir em 1 unidade, permanecer inalterado ou voltar para o mínimo. Uma nova solução é gerada através do algoritmo guloso (linha 7) e a solução obtida é submetida ao Iterated Greedy (linha 8). No final, a melhor solução encontrada é retornada.

4. Experimentos computacionais

Nesta seção os resultados obtidos são analisados e discutidos. Além disso, são apresentadas as instâncias, as configurações dos experimentos, parâmetros e métricas de avaliação.

4.1. Instâncias

Nos experimentos foi considerado um conjunto com 81 instâncias, classificadas como "Clusterizadas - C" (24 instâncias), "Randomizadas - R" (33 instâncias) e "Randomizadas Clusterizadas - RC" (24 instâncias), obtidas de Solomon [1987]. As instâncias variam em grupos de 25, 50 e 100 clientes. Em Faria et al. [2022], é feito um tratamento sobre essas instâncias e, para que seja

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

possível uma comparação entre os algoritmos desenvolvidos, o tratamento realizado neste trabalho é o mesmo. Dessa forma, assume-se que 80% dos clientes podem ser atendidos por drones, sendo eles os com menor demanda. O custo de transporte por unidade é fixado em 25 para os caminhões e 1 para os drones, sendo utilizada a métrica de Manhattan para calcular a distância para os caminhões e Euclidiana para os drones. O tempo de espera máximo de drones é definido como 10 unidades de tempo. Em relação à capacidade de carga, foi mantido o valor presente na instância original.

Para todas as instâncias, considera-se a velocidade de caminhões como sendo de 1 unidade de distância por unidade de tempo, enquanto a velocidade de drones é o dobro da de caminhões. Além disso, o tempo de atendimento do cliente quando atendido por drone é a metade do tempo de atendimento quando este é atendido por caminhão. Por último, cada drone pode atender um único cliente a cada decolagem.

4.2. Ambiente de teste e configuração do experimento

O algoritmo proposto neste trabalho foi executado em uma máquina com processador AMD Ryzen 5 5500U, 8 GB de memória RAM com 3200 MHz, sistema operacional Ubuntu 22.04.1 LTS x86_64 com kernel Linux 5.15.0-58-generic. Cada uma das 81 instâncias foi executada 30 vezes e foram aferidos o tempo de execução e a qualidade da solução encontrada.

Os parâmetros utilizados foram ajustados conforme testes realizados com o *framework Iterated Race* (IRACE) de López-Ibáñez et al. [2016] tomando como entrada um conjunto de 18 instâncias representativas das instâncias de entrada. Os valores de α, β, $nIter$ e $maxIterIG$ calibrados pelo $framework$ foram: $\alpha = 0.6$, $\beta = 0.3$ $nIter = 20$ e $maxIterIG = 50$.

4.3. Resultados e análise

Os resultados obtidos pela abordagem proposta são apresentados nas tabelas e comparados quanto ao custo e tempo de processamento com a abordagem MSPTH2, apresentada em Di Puglia Pugliese et al. [2021] e implementada por Faria [2022] utilizando parâmetros empíricos, e com o algoritmo MIP de Faria [2022].

As Tabelas 1, 2 e 3 apresentam a comparação dos resultados da abordagem MIG proposta com a heurística de duas fases MSPTH2 de Di Puglia Pugliese et al. [2021] e o GRASP com MIP apresentados em Faria [2022] para as instâncias com 25, 50 e 100 clientes, respectivamente. A coluna **Melhor Solução** traz o valor da melhor solução obtida dentre as três abordagens e serve de referência para o cálculo do desvio percentual, apresentado em relação à melhor solução obtida no experimento (coluna **Melhor**) e em relação à média das soluções obtidas (coluna **Média**). A coluna **T(s)** mostra o tempo em segundos. Os tempos dos três algoritmos foram convertidos para um mesmo modelo de ambiente computacional.

Na Tabela 1 pode-se observar que, para as instâncias de 25 clientes, o algoritmo MIG proposto obteve os melhores resultados para seis das 27 instâncias tanto em relação à diferença percentual da melhor solução quanto sobre a média das soluções. Destaca-se neste experimento que a heurística de duas fases MSPTH2 apresentou os melhores resultados quanto ao tempo. Quando comparado com o GRASP com MIP (coluna **MIP**), o MIG apresenta uma melhor eficiência de tempo, com uma diferença inferior a 10% em qualidade da solução.

Já na Tabela 2, para instâncias com 50 clientes, verifica-se que o algoritmo proposto é o que menos impacta no tempo, o que mostra a escalabilidade da abordagem. Além disso, observa-se que o algoritmo MIG obteve o melhor resultado em 10 das 27 instâncias e obteve a melhor média geral em relação à diferença percentual para a melhor solução e o menor valor médio de tempo.

Os resultados apresentados para as instâncias com 100 clientes, apresentados na Tabela 3, reforçam a escalabilidade do algoritmo proposto, tendo apresentado tempo médio quase 20% do

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

instância	Melhor	MIP			MSPTH2			MIG		
	Solução	Melhor	Média	T(s)	Melhor	Média	T(s)	Melhor	Média	T(s)
C201_025	4.970,9	0,123	0,123	0,666	0,155	0,162	**0,093**	**0,000**	**0,001**	0,346
C202_025	4.781,5	0,158	0,177	0,791	0,090	0,143	**0,080**	**0,000**	**0,000**	0,356
C203_025	4.638,0	0,018	0,120	0,943	**0,000**	**0,020**	**0,198**	0,030	0,031	0,401
C204_025	3.966,0	0,007	0,044	1,025	**0,000**	0,020	**0,133**	0,012	**0,015**	0,499
C205_025	4.606,0	**0,000**	**0,042**	0,730	0,043	0,058	**0,109**	0,075	0,076	0,316
C206_025	4.388,0	**0,000**	**0,004**	0,756	0,005	0,038	**0,090**	0,123	0,126	0,290
C207_025	4.348,0	0,004	0,040	0,929	**0,000**	**0,031**	**0,091**	0,125	0,130	0,339
C208_025	4.361,0	0,001	**0,016**	0,829	**0,000**	**0,016**	**0,096**	0,119	0,126	0,315
R201_025	7.049,5	0,295	0,383	0,871	0,341	0,368	**0,172**	**0,000**	**0,028**	0,189
R202_025	7.308,0	**0,000**	0,055	1,017	0,015	**0,043**	**0,230**	0,111	0,112	0,373
R203_025	7.020,0	0,024	0,047	1,122	0,022	0,123	**0,223**	**0,000**	**0,004**	0,415
R204_025	6.281,0	**0,000**	**0,042**	1,275	0,063	0,159	**0,242**	0,174	0,209	0,543
R205_025	6.070,0	0,001	0,092	0,935	**0,000**	**0,053**	**0,161**	0,089	0,128	0,290
R206_025	5.948,0	**0,000**	**0,042**	1,017	0,019	0,104	**0,220**	0,189	0,242	0,408
R207_025	5.888,0	**0,000**	**0,060**	1,125	0,033	0,182	**0,182**	0,079	0,091	0,416
R208_025	5.331,0	**0,000**	**0,074**	1,230	0,168	0,309	**0,186**	0,307	0,329	0,523
R209_025	5.921,6	0,027	**0,030**	0,799	0,029	0,035	**0,100**	**0,000**	0,058	0,348
R210_025	6.545,8	0,138	0,184	0,987	0,075	0,175	**0,279**	**0,000**	**0,000**	0,334
R211_025	5.079,0	**0,000**	**0,035**	0,900	0,072	0,199	**0,133**	0,045	0,050	0,339
RC201_025	8.169,0	0,009	0,042	0,804	**0,000**	**0,023**	**0,192**	0,055	0,055	0,309
RC202_025	6.664,0	0,044	0,124	0,962	**0,000**	**0,074**	**0,157**	0,186	0,255	0,354
RC203_025	6.682,0	0,080	**0,148**	0,977	**0,000**	0,152	**0,222**	0,208	0,211	0,406
RC204_025	6.250,0	**0,000**	**0,010**	1,025	0,001	0,052	**0,200**	0,034	0,034	0,554
RC205_025	7.963,0	**0,000**	**0,009**	0,809	0,014	0,041	**0,125**	0,052	0,052	0,357
RC206_025	6.539,0	**0,000**	**0,050**	0,860	0,011	0,118	**0,194**	0,222	0,222	0,357
RC207_025	6.411,0	**0,000**	**0,030**	0,877	0,032	0,091	**0,166**	0,039	0,042	0,367
RC208_025	5.603,0	0,026	**0,038**	0,834	**0,000**	0,040	**0,115**	0,008	0,073	0,394
Média	5.880,8	**0,035**	**0,076**	0,929	0,044	0,105	**0,163**	0,085	0,100	0,375

Tabela 1: Resultados para instâncias com 25 clientes para os algoritmos

algoritmo MSPTH2 e quase 11% do algoritmo GRASP-MIP, sendo que o algoritmo continua competitivo em termos de qualidade das soluções obtidas.

Nas tabelas 2 e 3, observa-se que o tempo de execução médio do algoritmo MIG foi consideravelmente menor para a maioria das instâncias em comparação com as outras técnicas utilizadas. Nota-se que para instâncias grandes, de 100 clientes, o tempo foi mais de 4 vezes menor em relação ao MSPTH2, porém, como pode ser observado na tabela 1, o tempo é um pouco maior em relação a esse mesmo algoritmo. Dessa forma, o algoritmo MIG se torna mais vantajoso à medida que o número de clientes aumenta.

5. Conclusão

Para solucionar o problema de roteamento de veículos capacitados com drones e janela de tempo (VRP-DTW), este trabalho propôs um *Multi-Start Iterated Greedy* (MIG) combinado com um *Random Variable Neighborhood Descend* (RVND). A partir de uma solução gerada pelo algoritmo guloso randomizado, o MIG aplica etapas de remoção parcial de clientes para depois reinserí-los, potencialmente melhorando a solução anterior. A partir dessa solução, o RVND busca melhorar a qualidade e foi implementado na forma como proposto por Faria [2022], com a adição de um movimento *2-opt*. O algoritmo implementado foi testado para um conjunto de 81 instâncias, comparado com os resultados obtidos pelo MSPTH2 e pelo MIP, de Faria [2022]. Em resumo, o algoritmo proposto demonstrou bons resultados, especialmente para as instâncias de 50 e 100 clientes em tempo de execução e qualidade. Também nota-se uma diminuição significativa para o tempo de execução, sendo cerca de 4 vezes mais rápido para as instâncias maiores, de 100 clientes.

Tendo em vista a eficiência em relação ao tempo de processamento nos resultados preliminares, futuramente pretende-se analisar novos parâmetros de forma a permitir que o algoritmo

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

instância	Melhor	MIP			MSPTH2			MIG		
	Solução	Melhor	Média	T(s)	Melhor	Média	T(s)	Melhor	Média	T(s)
C201_050	7.732,9	0,192	0,234	2,811	0,181	0,212	2,090	**0,000**	**0,067**	**0,970**
C202_050	7.408,1	0,141	0,203	3,559	0,173	0,212	2,531	**0,000**	**0,093**	**1,263**
C203_050	7.551,5	0,062	0,132	4,754	0,124	0,168	2,856	**0,000**	**0,089**	**1,498**
C204_050	7.379,0	**0,000**	0,085	7,292	0,017	0,101	2,513	0,010	**0,058**	**2,097**
C205_050	7.712,1	0,093	0,153	3,081	0,063	**0,067**	2,146	**0,000**	0,094	**1,087**
C206_050	7.808,0	**0,000**	0,050	3,673	0,025	**0,032**	2,321	0,009	0,070	**1,222**
C207_050	7.080,1	0,079	0,166	4,544	0,141	0,173	2,125	**0,000**	**0,134**	**1,271**
C208_050	7.728,9	0,014	0,057	4,111	0,036	**0,046**	3,036	**0,000**	0,069	**1,240**
R201_050	10.794,3	0,241	0,297	6,064	0,185	0,262	1,177	**0,000**	**0,006**	**0,830**
R202_050	10.435,7	0,116	0,157	4,624	0,015	0,144	1,609	**0,000**	**0,014**	**1,222**
R203_050	9.364,0	0,134	0,213	5,918	**0,000**	**0,110**	2,012	0,030	0,145	**1,485**
R204_050	8.215,8	0,029	0,083	6,260	0,045	0,131	**1,580**	**0,000**	**0,069**	1,591
R205_050	10.302,0	**0,000**	0,045	3,790	0,007	0,035	**0,910**	0,013	**0,028**	1,059
R206_050	9.222,0	0,020	0,109	4,850	**0,000**	**0,078**	1,394	0,051	0,165	**1,144**
R207_050	8.634,0	0,070	0,140	5,574	**0,000**	**0,101**	1,721	0,060	0,234	**1,387**
R208_050	8.067,0	0,053	0,089	6,054	**0,000**	**0,049**	1,605	0,020	0,143	**1,469**
R209_050	9.810,0	0,049	0,102	4,176	**0,000**	**0,037**	1,390	0,042	0,102	**1,199**
R210_050	8.834,0	0,168	0,222	4,433	**0,000**	**0,128**	1,715	0,016	0,135	**1,281**
R211_050	7.899,0	0,035	0,063	4,313	**0,000**	**0,045**	**0,794**	0,017	0,190	1,204
RC201_050	14.314,0	0,023	**0,086**	3,793	**0,000**	0,090	1,756	0,003	0,189	**0,864**
RC202_050	13.226,0	0,038	**0,064**	4,005	**0,000**	0,078	2,642	0,058	0,229	**1,120**
RC203_050	12.772,0	0,014	**0,028**	4,469	**0,000**	0,053	2,744	0,106	0,284	**1,311**
RC204_050	10.478,4	0,037	0,076	4,915	0,080	0,141	**1,414**	**0,000**	**0,059**	2,143
RC205_050	13.219,0	**0,000**	**0,021**	3,677	0,017	0,064	2,170	0,049	0,130	**1,129**
RC206_050	12.434,0	0,035	**0,076**	3,769	**0,000**	0,131	1,829	0,057	0,343	**1,069**
RC207_050	12.011,0	0,014	**0,082**	3,708	**0,000**	0,154	1,890	0,028	0,245	**1,207**
RC208_050	10.380,0	**0,000**	**0,069**	3,553	0,024	0,080	1,486	0,002	0,166	**1,264**
Média	9.659,7	0,061	0,115	4,510	0,042	**0,108**	1,906	**0,021**	0,132	**1,282**

Tabela 2: Resultados para instâncias com 50 clientes para os algoritmos

instância	Melhor	MIP			MSPTH2			MIG		
	Solução	Melhor	Média	T(s)	Melhor	Média	T(s)	Melhor	Média	T(s)
C201_100	13.707,0	0,120	0,301	19,988	**0,000**	**0,012**	16,601	0,085	0,263	**3,630**
C202_100	13.614,0	0,054	0,232	28,715	**0,000**	**0,012**	17,904	0,132	0,292	**4,789**
C204_100	14.506,0	**0,000**	**0,068**	67,851	0,124	0,210	61,433	0,066	0,089	**8,331**
C205_100	13.361,0	0,034	0,106	22,564	**0,000**	**0,051**	36,288	0,233	0,360	**4,319**
C206_100	13.032,0	0,060	0,132	27,401	**0,000**	**0,045**	38,027	0,159	0,468	**4,737**
C207_100	13.185,0	0,080	0,191	30,993	**0,000**	**0,078**	36,529	0,288	0,524	**4,606**
C208_100	13.010,0	0,044	0,188	32,083	**0,000**	**0,115**	41,318	0,211	0,327	**4,862**
R201_100	14.716,3	0,124	0,177	22,397	0,106	0,174	8,979	**0,000**	**0,000**	**3,286**
R202_100	14.103,8	0,067	0,169	25,977	0,052	0,130	15,301	**0,000**	**0,000**	**4,666**
R203_100	13.683,8	0,118	0,211	35,473	0,066	0,136	22,975	**0,000**	**0,003**	**6,345**
R204_100	11.139,0	0,161	0,259	52,583	**0,000**	**0,058**	23,342	0,143	0,163	**7,233**
R205_100	14.066,0	0,060	0,127	25,608	**0,000**	**0,014**	9,594	0,153	0,153	**4,104**
R206_100	13.460,3	0,021	0,142	30,564	0,046	0,114	16,048	**0,000**	**0,004**	**5,194**
R207_100	12.240,0	0,158	0,226	37,496	**0,000**	0,131	23,801	0,067	**0,067**	**6,286**
R208_100	10.652,0	0,238	0,314	57,682	**0,000**	**0,021**	21,504	0,159	0,159	**7,072**
R209_100	14.201,0	0,046	0,095	26,569	**0,000**	**0,017**	11,077	0,118	0,118	**4,995**
R210_100	13.379,8	0,091	0,163	28,001	0,008	0,064	20,092	**0,000**	**0,000**	**5,218**
R211_100	11.530,0	0,127	0,191	33,495	**0,000**	**0,011**	10,190	0,046	0,046	**5,244**
RC201_100	22.324,0	0,117	0,217	229,104	**0,000**	**0,100**	13,755	0,136	0,136	**3,220**
RC202_100	19.616,0	0,046	0,163	44,108	**0,000**	0,099	25,528	0,098	**0,098**	**4,510**
RC203_100	18.053,2	0,039	0,099	33,111	0,059	0,116	28,970	**0,000**	**0,000**	**6,065**
RC204_100	16.687,0	0,027	0,075	47,009	**0,000**	0,111	22,204	0,003	**0,003**	**7,495**
RC205_100	22.542,4	0,059	0,125	170,022	0,016	**0,065**	19,796	**0,000**	0,137	**4,005**
RC206_100	20.232,6	0,011	0,088	25,893	0,034	0,158	11,100	**0,000**	**0,000**	**4,159**
RC207_100	18.046,1	**0,000**	**0,073**	25,231	0,063	0,167	17,657	0,119	0,119	**4,747**
RC208_100	15.712,7	0,111	0,181	30,865	0,116	0,175	12,220	**0,000**	**0,000**	**4,827**
Média	14.997,1	0,075	0,164	46,326	**0,029**	**0,093**	23,097	0,086	0,141	**5,195**

Tabela 3: Resultados para instâncias com 100 clientes para os algoritmos

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br

utilize essa margem extra para encontrar soluções com um custo ainda menor. Além disso, pretende-se analisar a vantagem de se aplicar o *2-opt* no RVND para refinar a solução.

Referências

Chiang, W.-C., Li, Y., Shang, J., e Urban, T. L. (2019). Impact of drone delivery on sustainability and cost: Realizing the uav potential through vehicle routing optimization. *Applied Energy*, 242: 1164–1175. ISSN 0306-2619.

Di Puglia Pugliese, L., Macrina, G., e Guerriero, F. (2021). Trucks and drones cooperation in the last-mile delivery process. *Networks*, 78(4):371–399.

Euchi, J. e Sadok, A. (2021). Hybrid genetic-sweep algorithm to solve the vehicle routing problem with drones. *Physical Communication*, 44:101236. ISSN 1874-4907.

Faria, E. B. R. (2022). *Heurísticas para o Problema de Roteamento de Veículos com Drones e Janelas de Tempo*. Monografia, Universidade Federal de Juiz de Fora, Juiz de Fora.

Faria, E. B. R., Soares, S. S. R. F., Gonçalves, L. B., e Moreno, L. L. O. (2022). Um algoritmo híbrido para o problema de roteamento de caminhões e drones com janela de tempo. In *Simpósio Brasileiro de Pesquisa Operacional (SBPO)*, volume 54, Juiz de Fora.

Lei, D., Cui, Z., e Li, M. (2022). A dynamical artificial bee colony for vehicle routing problem with drones. *Engineering Applications of Artificial Intelligence*, 107:104510.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., e Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58. ISSN 2214-7160.

Murray, C. C. e Chu, A. G. (2015). The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54: 86–109. ISSN 0968-090X.

Sacramento, D., Pisinger, D., e Ropke, S. (2019). An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transportation Research Part C: Emerging Technologies*, 102:289–315. ISSN 0968-090X.

Schermer, D., Moeini, M., e Wendt, O. (2018). A variable neighborhood search algorithm for solving the vehicle routing problem with drones. *Technacal Report*.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.

Wang, X., Poikonen, S., e Golden, B. (2017). The vehicle routing problem with drones: several worst-case results. *Optimization Letters*, 11(4):679–697. ISSN 1862-4480.

Wang, Z. e Sheu, J.-B. (2019). Vehicle routing problem with drones. *Transportation Research Part B: Methodological*, 122:350–364. ISSN 0191-2615.

DOI: 10.59254/sbpo-2023-175209
https://proceedings.science/p/175209?lang=pt-br