

# Problema de roteamento de veículos verdes multi-depósito



## **A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem**

**Nomes:** Halliday Gauss, Emanuel Xavier

**Matrícula:** 18.4093, 18.1.4148

# O Problema



**O Multi-Depot Green Vehicle Routing Problem (MDGVRP) é uma extensão do bem conhecido Green Vehicle Routing Problem (GVRP), onde uma frota de veículos movidos a combustível alternativo (AFVs) são usados para atender os clientes. No MDGVRP, os AFVs são despachados de diferentes locais de depósito e podem reabastecer durante o dia em qualquer depósito ou estação de reabastecimento.**

## O Problema - Formulação Matemática



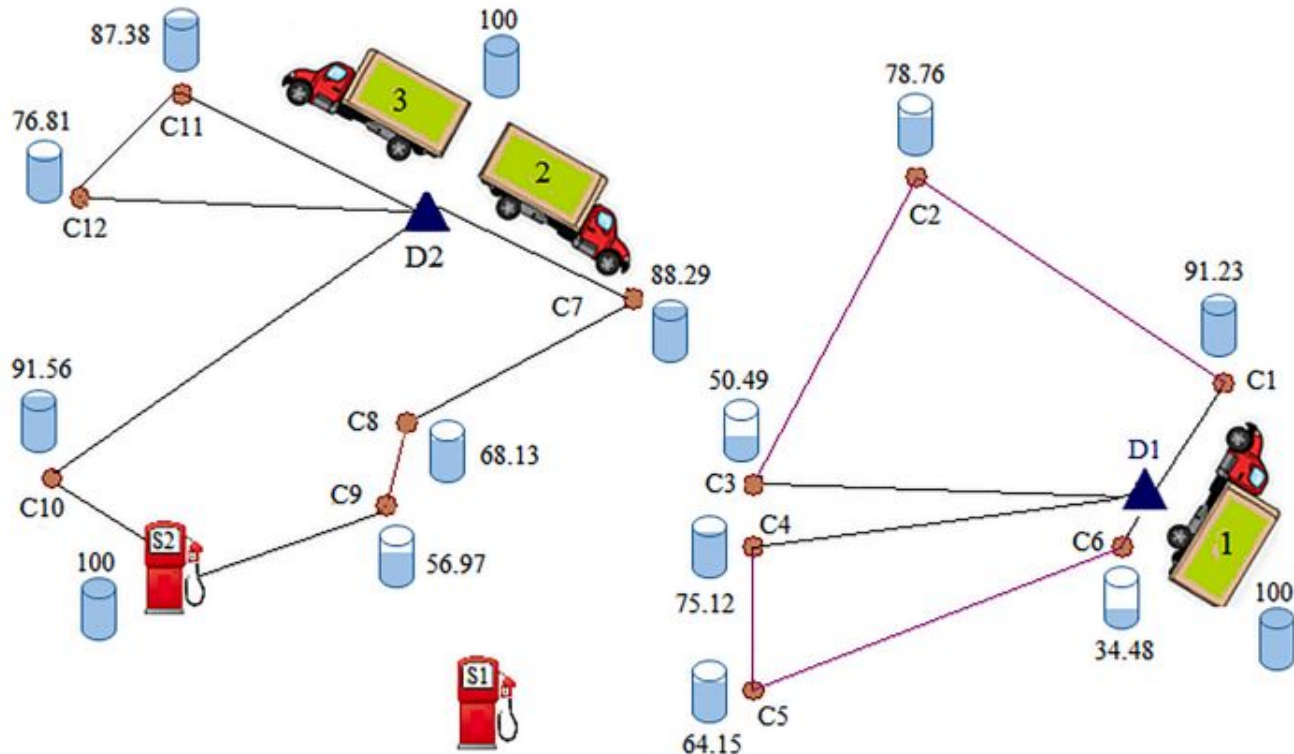
Seja  $G = (V, A)$  um grafo direcionado completo, onde  $V$  e  $A$  representam o conjunto de vértices e o conjunto de arcos, respectivamente.  $V$  consiste em três subconjuntos: conjunto de depósitos  $D = \{v_1, v_2, \dots, v_m\}$ , conjunto de clientes  $I = \{v_{m+1}, v_{m+2}, \dots, v_{m+n}\}$ , e um conjunto de postos de reabastecimento  $F = \{v_{m+n+1}, v_{m+n+2}, \dots, v_{m+n+f}\}$ .  $s_i$  é o tempo de atendimento (tempo de reabastecimento) no cliente (estação)  $i$ . Supondo que o tanque de combustível do veículo está totalmente cheio e o tempo de reabastecimento é constante. Um custo não negativo  $d_{ij}$  e tempo de viagem  $t_{ij}$  estão associados a cada arco  $(i, j)$ . A frota é composta por veículos idênticos com taxa de consumo de combustível  $r$  e capacidade do tanque de combustível  $Q$ . Os veículos partem e retornam ao mesmo depósito.

## O Problema - Formulação Matemática




O objetivo é desenhar um conjunto de rotas que visitem cada cliente uma vez de forma que a distância total percorrida seja minimizada. Uma duração máxima da rota  $T_{max}$  é imposta ao tempo de viagem dos veículos. Uma estação de reabastecimento pode ser visitada várias vezes pelo mesmo veículo ou por veículos diferentes. Portanto, para cada visita potencial, foi adicionado uma cópia da estação e criado  $G' = (V', A')$ , Onde  $V' = V \cup F'$  e  $F' = F \cup \{v_{m+n+f+1}, v_{m+n+f+2}, \dots, v_{m+n+f+f'}\}$

# O Problema -Exemplo



## Sets - Conjuntos

- 
- I** -> Conjunto de Clientes.
  - D** -> Conjunto de Depósitos.
  - F** -> Conjunto de postos de abastecimentos.
  - V** -> Conjunto de Depósitos, Cliente, e postos de abastecimento  $\{V = I \cup D \cup F\}$ .
  - F'** -> Conjunto de todos os postos de abastecimentos incluindo os fictícios.
  - R** -> Conjunto de todos os clientes e postos de abastecimentos.  $\{R = I \cup F'\}$
  - V'** -> Conjunto de todos os vértices  $\{V' = R \cup D\}$

## Parâmetros



$d_{ij}$  -> Distância do nó  $i$  para o nó  $j$ .

$t_{ij}$  -> Tempo de viagem do nó  $i$  para o nó  $j$ .

$Q$  -> Capacidade do tanque de combustível dos veículos.

$r$  -> Taxa de consumo de combustível do veículo.

$S_i$  -> Tempo de descarga ou abastecimento no nó  $i$ .

$T_{Max}$  -> Duração máxima da viagem.

$M$  -> Constante grande.

## Variáveis de Decisão



**$X_{ij}$  -> 1 se nó  $j$  é visitado por um veículo após passar pelo nó  $i$ . o caso contrário.**

**$Z_{ij}$  -> 1 se nó  $i$  é visitado por um veículo responsável pelo depósito  $j$ . o caso contrário**

**$Y_i$  -> Combustível do veículo após chegar no nó  $i$ . Obs : ( $Y_i = Q$ , se  $i$  está em  $F'$ )**

**$T_i$  -> Hora de chegada no nó  $i$ .**




## Função Objetiva


$$\min \sum_{i \in V'} \sum_{\substack{j \in V \\ i \neq j}} d_{ij} x_{ij}$$

-> O objetivo é minimizar a distância percorrida pelos veículos

## Restrições


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

-> O objetivo dessa restrição é garantir que apenas um nó  $j$  seja visitado a partir de um nó  $i$  que é um cliente.

$$\sum_{\substack{j \in V \\ i \neq j}} x_{ji} = 1, \quad \forall i \in I$$

-> O objetivo dessa restrição é garantir que um cliente  $i$  seja visitado a partir de um único nó  $j$ .

-> Juntas essas restrições garantem que cada cliente seja visitado uma única vez.

## Restrições

$$\sum_{j \in D} z_{ij} = 1,$$

$$\forall i \in I$$


-> O objetivo dessa restrição é garantir que um cliente  $i$  esteja associado a um único depósito  $j$ .

$$\sum_{\substack{j \in V' \\ i \neq j}} x_{ij} \leq 1,$$

$$\forall i \in F'$$

-> O objetivo dessa função é garantir que um posto  $i$  seja visitado no máximo uma vez. Obs: Como o somatório é  $\leq 1$ , então não é obrigatório passar em um posto.

# Restrições


$$\sum_{\substack{i \in V' \\ i \neq j}} x_{ij} - \sum_{\substack{i \in V' \\ i \neq j}} x_{ji} = 0,$$

$$\forall j \in V'$$

-> O objetivo dessa restrição é garantir o controle de fluxo, ou seja, todo nó deve ter uma aresta de entrada e uma de saída.

## Restrições

$$x_{ij} \leq z_{ij},$$

$$\forall i \in R, \forall j \in D$$

-> O objetivo dessas restrições é garantir que um veículo viaje a partir de um nó  $i$  para um depósito  $j$  ou um depósito  $j$  para o nó  $i$  somente se esse nó  $i$  estiver associado com o depósito  $j$ .

$$x_{ji} \leq z_{ij},$$

$$\forall i \in R, \forall j \in D$$

## Restrições


$$x_{ik} + z_{ij} - z_{kj} \leq 1, \quad \forall i \in R, \forall k \in R, i \neq k, \forall j \in D$$

$$x_{ik} + z_{kj} - z_{ij} \leq 1, \quad \forall i \in R, \forall k \in R, i \neq k, \forall j \in D$$

-> O objetivo dessas restrições é garantir que dois nós (Cliente ou Posto), que foram visitados consecutivamente pelo mesmo veículo, devem ser atribuídos a uma mesmo depósito.


## Restrições

$$\tau_j \geq \tau_i + s_i + t_{ij} - M(1 - x_{ij}), \quad \forall i \in V', \forall j \in R, i \neq j$$

$$\tau_j \leq \tau_i + s_i + t_{ij} + M(1 - x_{ij}), \quad \forall i \in V', \forall j \in R, i \neq j$$

-> O objetivo dessas restrições é garantir que, se existir a aresta de  $i$  para  $j$ , sendo  $i$  e  $j$  pertencentes aos clientes ou postos, então a Hora de Chegada na cidade  $j$  tem que ser exatamente a Hora de chegada na cidade  $i$  somado com o tempo gasto para fazer o abastecimento ou descarga, mais o tempo de viagem da cidade  $i$  para a cidade  $j$ .


## Restrições


$$\tau_i + s_i + t_{ij}x_{ij} \leq T_{max} ,$$

$\forall i \in R, \forall j \in D$  -> O objetivo dessa restrição é garantir o tempo gasto no procedimento de rota não seja maior que o tempo TMax estipulado.




## Restrições


$$y_j \leq y_i - (r \cdot d_{ij})x_{ij} + Q(1 - x_{ij}), \quad \forall i \in V, \forall j \in I, i \neq j$$

-> O objetivo dessa restrição é garantir que o controle do combustível do tanque em cada Cliente. Ou seja, o combustível até a cidade  $j$ , dado que tem aresta escolhida de  $i$  para  $j$ , tem que ser menor ou igual ao combustível que o veículo tinha na cidade  $i$  menos o combustível gasto para chegar na cidade  $j$ .

## Restrições


$$y_i \geq \sum_{\substack{i \in V' \\ i \neq j}} (r \cdot d_{ij}) x_{ij},$$

$$\forall i \in I$$

-> O objetivo dessa restrição é garantir que o veículo nunca fique sem combustível. Ou seja, a gasolina em um nó Cliente  $i$ , tem que ser maior ou igual ao gasto que se terá ao ir para uma cidade  $j$ .

$$y_i = Q, \quad \forall i \in F'$$

-> O objetivo dessa restrição é garantir que o veículo fique de tanque cheio ao visitar um posto de abastecimento.

# Restrições

$x_{ij} \in \{0, 1\}, \quad \forall i, j \in V'$       -> Restrição de Domínio  $X_{ij}$

$z_{ij} \in \{0, 1\}, \quad \forall i \in R, \forall j \in D$       -> Restrição de Domínio  $Z_{ij}$

$y_i \geq 0, \quad \forall i \in V'$       -> Restrição de Não Negatividade em  $Y_i$   
(Combustível do veículo no nó  $i$ ).

$\tau_i \geq 0, \quad \forall i \in V'$       -> Restrição de Não Negatividade em  $T_i$   
(Hora de Chegada do veículo no nó  $i$ ).

# Implementação

## #conjuntos

set  $I := \{1..12\}$ ; # -> Conjunto de Clientes.

set  $D := \{13..14\}$ ; # -> Conjunto de Depósitos.

set  $F := \{15\}$ ; # -> Conjunto de postos de abastecimentos.

set  $V := I \cup D \cup F$ ; # -> Conjunto de Depósitos, Cliente, e postos de abastecimento  $\{V = I \cup D \cup F\}$ .

set  $F_ := F \cup \{16\}$ ; # -> Conjunto de todos os postos de abastecimentos incluindo os fictícios.

set  $R := I \cup F_$ ; # -> Conjunto de todos os clientes e postos de abastecimentos.  $\{R = I \cup F\}$

set  $V_ := R \cup D$ ; # -> Conjunto de todos os vértices  $\{V' = R \cup D\}$

## #parametros

param  $d \{i \in V_ , j \in V_ \}$ ; # -> Distância do nó  $i$  para o nó  $j$ .

param  $t \{i \in V_ , j \in V_ \}$ ; # -> Tempo de viagem do nó  $i$  para o nó  $j$ .

param  $Q$ ; # -> Capacidade do tanque de combustível dos veículos.

param  $r$ ; # -> Taxa de consumo de combustível do veículo..

param  $S \{i \in V_ \}$ ; # -> Tempo de descarga ou abastecimento no nó  $i$ .

param  $TMax$ ; # -> Duração máxima da viagem.

param  $M$ ; # -> Constante grande.

# Implementação

#variaveis

var X {i in V\_, j in V\_} >= 0, binary; #-> 1 se nó j é visitado por um veículo após passar pelo nó i. O caso contrário.

var Z {i in R, j in D} >= 0, binary; #-> 1 se nó i é visitado por um veículo responsável pelo depósito j. O caso contrário

var Y {i in V\_} >= 0; #-> Combustível do veículo após chegar no nó i. Obs: ( $\forall i = Q$ , se i está em F)

var T {i in V\_} >= 0; #-> Hora de chegada no nó i.

#funcao objetiva

#-> O objetivo é minimizar a distância percorrida pelos veículos

minimize distancia : sum {i in V\_} (sum {j in V\_} if i != j then d[i,j]\*X[i,j]);

# Implementação

## #restricoes

#-> O objetivo dessa restrição é garantir que apenas um nó  $j$  seja visitado a partir de um nó  $i$  que é um cliente.

```
- r1 {i in I}: sum{j in V_} if i != j then X[i, j] = 1;
```

#-> O objetivo dessa restrição é garantir que um cliente  $i$  seja visitado a partir de um único nó  $j$ .

```
- r2 {i in I}: sum{j in V_} if i != j then X[j, i] = 1;
```

#-> Juntas essas restrições ( $r1$  e  $r2$ ) garantem que cada cliente seja visitado uma única vez.

#-> O objetivo dessa restrição é garantir que um cliente  $i$  esteja associado a um único depósito  $j$ .

```
r3 {i in I}: sum{j in D} Z[i, j] = 1;
```

#-> O objetivo dessa função é garantir que um posto  $i$  seja visitado no máximo uma vez. Obs: Como o somatório é  $\leq 1$ , então não é obrigatório passar em um posto.

```
- r4 {i in F_}: sum{j in V_} if i != j then X[i, j] <= 1;
```

#-> O objetivo dessa restrição é garantir o controle de fluxo, ou seja, todo nó deve ter uma aresta de entrada e uma de saída.

```
- r5 {j in V_}: (sum{i in V_} if i != j then X[i, j]) - (sum{i in V_} if i != j then X[j, i]) = 0;
```

#-> O objetivo dessas restrições ( $r6$  e  $r7$ ) é garantir que um veículo  $i$  viaje a partir de um nó  $i$  para um depósito  $j$  ou um depósito  $j$  para o nó  $i$  somente se esse o nó  $i$  estiver associado com o depósito  $j$ .

```
r6 {i in R, j in D}: X[i, j] <= Z[i, j];
```

```
r7 {i in R, j in D}: X[j, i] <= Z[i, j];
```

#-> O objetivo dessas restrições ( $r8$  e  $r9$ ) é garantir que dois nós (Cliente ou Posto), que foram visitados consecutivamente pelo mesmo veículo, devem ser atribuídos a uma mesmo depósito.

```
- r8 {i in R, k in R, j in D}: if i != k then X[i, k] + Z[i, j] - Z[k, j] <= 1;
```

```
- r9 {i in R, k in R, j in D}: if i != k then X[i, k] + Z[k, j] - Z[i, j] <= 1;
```

# Implementação

# O objetivo dessas restrições (r10 r11) é garantir que, se existir a aresta de  $i$  para  $j$ , sendo  $i$  e  $j$  pertencentes aos clientes ou postos, então a Hora de Chegada na cidade  $j$  tem que ser exatamente a Hora de chegada na cidade  $i$  somado com o tempo gasto para fazer o abastecimento ou descarga, mais o tempo de viagem da cidade  $i$  para a cidade  $j$ .

```
r10 {i in V_, j in R}: if i != j then T[j] >= T[i] + S[i] + t[i,j] - M*(1 - X[i,j]);  
r11 {i in V_, j in R}: if i != j then T[j] <= T[i] + S[i] + t[i,j] + M*(1 - X[i,j]);
```

#-> O objetivo dessa restrição é garantir o tempo gasto no procedimento de rota não seja maior que o tempo TMax estipulado.

```
r12 {i in R, j in D}: T[i] + S[i] + t[i,j]*X[i,j] <= TMax;
```

#-> O objetivo dessa restrição é garantir que o controle do combustível do tanque em cada Cliente. Ou seja, o combustível até a cidade  $j$ , dado que tem aresta escolhida de  $i$  para  $j$ , tem que ser menor ou igual ao combustível que o veículo tinha na cidade  $i$  menos o combustível gasto para chegar na cidade  $j$ .

```
r13 {i in V_, j in I}: if i != j then Y[j] <= Y[i] - (r*d[i,j])*X[i,j] + Q*(1-X[i,j]);
```

#-> O objetivo dessa restrição é garantir que o veículo nunca fique sem combustível. Ou seja, a gasolina em um nó Cliente  $i$ , tem que ser maior ou igual ao gasto que se terá ao ir para uma cidade  $j$ .

```
r14 {i in I}: Y[i] >= sum {j in V_} if i != j then r*d[i,j]*X[i,j];
```

#-> O objetivo dessa restrição é garantir que o veículo fique de tanque cheio ao visitar um posto de abastecimento.

```
r15 {i in F_ union D}: Y[i] = Q;
```

```
solve;
```



# Implementação

[illegible]



```
param t:
```

[illegible]

# Implementação

```
param Q := 100; # -> Capacidade do tanque de combustível dos veículos.  
  
param r := 1; # -> Taxa de consumo de combustível do veículo.  
  
param TMax := 200; # -> Duração máxima da viagem.  
  
# -> Tempo de descarga ou abastecimento no nó i.  
param S :=  
1 10  
2 11  
3 12  
4 13  
5 14  
6 15  
7 16  
8 17  
9 18  
10 19  
11 20  
12 21  
13 10  
14 10  
15 8  
16 8;
```

# Referências



**A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem -**

<https://sci-hub.ru/https://www.sciencedirect.com/science/article/pii/S1366554521000673>