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 MODELO EXATO PARA O ORIENTEERING PROBLEM COM SELEÇÃO DE HOTÉIS

João Marcos Pereira Silva
Programa de Pós-Graduação em Engenharia de Produção - Universidade Federal Fluminense
Rua Passo da Pátria, 156, Campus Praia Vermelha, Bloco D - sala 309, São Domingos, Niterói-RJ
joaomarcossilva@id.uff.br

Marcos Roboredo, Diogo Ferreira de Lima Silva, Eduardo Uchoa
Departamento de Engenharia de Produção - Universidade Federal Fluminense
Rua Passo da Pátria, 156, Campus Praia Vermelha, Bloco D - sala 306, São Domingos, Niterói-RJ
mcroboredo@id.uff.br, diogofls@id.uff.br, eduardo_uchoa@id.uff.br

RESUMO

O *Orienteering Problem* com seleção de hotéis (OPHS) considera um conjunto de hotéis e um conjunto de clientes, onde cada cliente possui uma pontuação pré-estabelecida. O problema visa determinar um número fixo de viagens conectadas, começando e terminando em um dos hotéis, limitadas em tempo, que maximize a soma das pontuações obtidas ao visitar os clientes. Para resolver o problema, propomos pela primeira vez um modelo exato, consistindo em um algoritmo de Branch-and-Cut-and-Price implementado com auxílio da ferramenta computacional VRPSolver. O diferencial do modelo proposto é que este considera variáveis para as viagens ao invés das jornadas completas. A concatenação das viagens é garantida através de variáveis e restrições que são incluídas no problema mestre. O modelo foi aplicado em instâncias propostas na literatura com até 102 nós, 8 hotéis e 4 viagens. Resultados mostram que o modelo encontrou a solução ótima para 165 dentre as 175 instâncias testadas.

PALAVRAS CHAVE. Problema de roteamento de veículos, Seleção de hotéis, Branch-cut-and-price, VRPSolver.

Tópicos (Otimização Combinatória)

ABSTRACT

The Orienteering Problem with Hotel Selection (OPHS) considers a set of hotels and a set of customers, where each customer has a pre-established score. The problem aims to determine a fixed number of connected trips starting and ending at one of the hotels, limited in time, that maximizes the sum of the scores obtained by visiting the customers. For the problem, we propose for the first time an exact model, which is a Branch-and-Cut-and-Price algorithm within the computational tool VRPSolver. The differential of the proposed model is that it considers variables for trips instead of complete journeys. The concatenation of trips is ensured through variables and constraints that are included in the master problem. The model is applied to instances proposed in the literature with up to 102 nodes, 8 hotels, and 4 trips. Results show that the model found the optimal solution for 165 out of 175 tested instances.

KEYWORDS. Vehicle routing problem, Hotel selection, Branch-cut-and-price, VRPSolver.

Paper topics (Combinatorial Optimization)

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

1. Introdução

No clássico *Orienteering Problem* (OP), um conjunto de vértices é fornecido, onde cada vértice possui uma pontuação associada. O objetivo é determinar um caminho limitado em tempo que visita um subconjunto de vértices e maximiza a soma das pontuações coletadas. O OP pode modelar várias aplicações reais em logística, turismo e outros campos. Para mais detalhes sobre variantes do OP, abordagens de solução e aplicações, consulte Vansteenwegen et al. [2011] e Gunawan et al. [2016].

O presente estudo lida com uma extensão do OP, denominada *Orienteering Problem* com Seleção de Hotéis (OPHS), apresentada inicialmente por Divsalar et al. [2013]. Além do conjunto original de vértices (que representa classicamente os clientes), a variação OPHS considera um conjunto de hotéis. O problema visa determinar um número fixo de viagens conectadas, limitadas em tempo, que maximize a soma das pontuações obtidas ao visitar os clientes. Cada viagem deve começar e terminar em um hotel. Além disso, o hotel em que a primeira viagem começa e o hotel em que a última viagem termina são pré-estabelecidos.

No melhor do nosso conhecimento, há seis artigos na literatura que tratam do OPHS e todos apresentaram abordagens heurísticas para resolver o problema. Divsalar et al. [2013] propuseram formalmente o OPHS e apresentaram duas contribuições: primeiro, os autores propuseram um método heurístico baseado em busca em vizinhança, segundo, um conjunto de 229 instâncias de referência, projetadas usando instâncias do OP de tamanhos variados. Em seguida, Divsalar et al. [2014] propuseram um algoritmo memético e um conjunto de 176 novas instâncias grandes para o OPHS. Em seguida, Toledo et al. [2019] apresentaram uma hiper-heurística para o OPHS. Recentemente, Sohrabi et al. [2020] e Sohrabi et al. [2021] propuseram algoritmos para o OPHS baseados em *Greedy Randomized Adaptive Search Procedure* (GRASP) e Sistema de Colônia de Formigas (ACS). Sohrabi et al. [2020] também apresentaram instâncias para um caso particular do problema onde o tamanho máximo para cada uma das viagens é sempre o mesmo. Finalmente, uma formulação biobjetivo para o OPHS foi proposta por Ataei et al. [2022], onde os autores consideraram a utilidade de passeios turísticos além de uma pontuação ponderada para a seleção de hotéis.

Conforme afirmado por Costa et al. [2019], as principais abordagens exatas para resolver problemas de roteamento de veículos são os algoritmos *branch-cut-and-price* (BCP). Por outro lado, os algoritmos BCP de última geração propostos para esses problemas envolvem várias técnicas avançadas para as quais a implementação é difícil e demorada até mesmo para uma equipe habilidosa. Para superar esse problema, Pessoa et al. [2020] propuseram o VRPSolver, um algoritmo BCP exato e genérico que incorpora muitos elementos estado da arte para algoritmos BCP, como cortes de Rank-1 com memória limitada, eliminação de arcos com base no custo reduzido, enumeração de rotas elementares, dentre outros que podem ser utilizados por meio de uma simples parametrização. Desde sua proposição [Pessoa et al., 2020], aplicações do VRPSolver foram apresentadas na literatura e este solucionador genérico mostrou resultados competitivos ou mesmo de fronteira para diferentes problemas de roteamento [Ben Mohamed et al., 2023; Clementino et al., 2022; Pessoa et al., 2021; Roboredo et al., 2023; Queiroga et al., 2021; Freitas et al., 2023].

O OPHS pode ser visto como um problema de roteamento com paradas intermediárias (os hotéis podem ser considerados como pontos intermediários de parada ao longo da rota). Em estudo recente, Roboredo et al. [2023] propuseram modelos do VRPSolver para esse tipo de problema, onde os autores consideram uma variável para representar cada rota viável completa. Porém, essa abordagem não é eficiente para resolver o OPHS, uma vez a jornada completa (cojunto de viagens conectadas) na solução ótima do OPHS tende a ser muito longa, o que sabidamente torna o

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

subproblema de *pricing* complexo.

Neste artigo, pela primeira vez, um modelo exato é proposto para o OPHS. Tal modelo é baseado em um algoritmo BCP implementado com auxílio da ferramenta VRPSolver. O grande diferencial do modelo proposto é que este considera uma variável para cada viagem ao invés de uma para cada jornada completa. A concatenação das viagens é garantida através de variáveis e restrições que são incluídas no problema mestre.

Visando comprovar a robustez do modelo proposto, o modelo proposto foi aplicado em instâncias da literatura com até 102 nós, 8 hotéis e 4 viagens. Resultados mostram que o modelo encontrou a solução ótima para 165 dentre as 175 instâncias testadas. Das 175 instâncias testadas, o modelo encontrou a solução ótima de 169 em até 3h de execução.

O restante deste artigo está organizado da seguinte forma. Na Seção 2, definimos formalmente o OPHS. Na Seção 3, apresentamos o modelo VRPSolver proposto, que é composto de grafos geradores de caminhos restritos em recurso e uma formulação de Programação Linear Inteira Mista (PLIM), que faz uso de variáveis associadas a caminhos nos grafos. A Seção 4 apresenta os resultados computacionais do modelo proposto. Finalmente, na Seção 5, resumimos nossas conclusões.

2. O Problema

Seja $G = (V, E)$ um grafo não direcionado, onde seu conjunto de vértices $V = V^+ \cup H$ é definido como a união de um conjunto de m hotéis ($H = \{1, ..., m\}$) e um conjunto de n clientes ($V^+ = \{m + 1, ..., m + n\}$). O conjunto de arestas E é definido como $E = \{(i, j)\} : i, j \in V, i < j\}$. Cada cliente $i \in V^+$ tem uma pontuação $s_i > 0$ associada, e cada aresta $e \in E$ é ponderada pelo tempo de viagem correspondente $t_e > 0$ (por conveniência, consideramos $t_{(j,i)} = t_{(i,j)}$, para todo $(i, j) \in E$).

Uma viagem é definida como uma sequência de nós visitados, iniciando e terminando em um hotel, de modo que quaisquer outros nós sejam clientes. Uma jornada é definida como uma sequência de viagens sem clientes comuns, onde a primeira viagem começa no hotel 1, a origem de qualquer outra viagem deve ser o fim da viagem anterior e a última viagem termina no hotel m. Além disso, cada cliente pode ser visitado no máximo uma vez durante a jornada. O objetivo é encontrar uma jornada que maximize a soma das pontuações dos clientes visitados, contendo exatamente D viagens, respeitando o tempo máximo de cada viagem k, dado por T_k, e o tempo máximo da jornada T_{max}.

Agora, propomos um exemplo ilustrativo de uma pequena instância do OPHS com $|H| = m = 3$, $|C| = n = 2$, $D = 2$, $T_1 = T_2 = 2$, $T_{max} = 4$, $s_4 = s_5 = 1$, $t_{\{1,4\}} = t_{\{2,4\}} = t_{\{2,5\}} = t_{\{3,5\}} = 1.0$ e $t_{\{1,2\}} = t_{\{2,3\}} = 1.5$. O tempo de viagem para cada uma das arestas restantes é de 5. A Figura 1 apresenta uma ilustração para esta instância proposta. Os vértices dos hotéis são marcados em amarelo, enquanto os vértices dos clientes são marcados em verde. A pontuação de cada cliente é indicada acima de seu vértice. Além disso, os tempos de viagem são indicados nas arestas. Omitimos as arestas cujo tempo de travessia é maior do que T_{max}, pois elas não podem pertencer a nenhuma solução ótima.

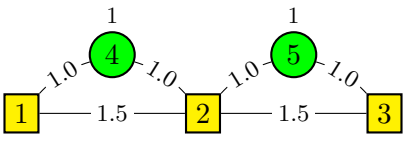


Figura 1: Ilustração do grafo para o exemplo ilustrativo.

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

No exemplo proposto, as seguintes jornadas são possíveis: $Tour_1 = (1, 2, 3)$, $Tour_2 = (1, 4, 2, 3)$, $Tour_3 = (1, 2, 5, 3)$, $Tour_4 = (1, 4, 2, 5, 3)$. O ótimo é alcançado com o $Tour_4$, composto pelas viagens $(1, 4, 2)$ e $(2, 5, 3)$, onde ambos os clientes são visitados, e a pontuação total obtida é $1 + 1 = 2$.

3. Modelo VRPSolver para o OPHS

Nesta seção, propomos um modelo VRPSolver para o OPHS. Um modelo VRPSolver pode ser entendido como um modelo PLIM que considera variáveis de caminho em um ou mais grafos direcionados. O modelo é resolvido pela ferramenta através de um algoritmo BCP genérico, onde as variáveis de caminho são geradas sob demanda pela resolução do subproblema de *pricing* que é modelado como o Problema do Caminho Mais Curto com Restrição de Recursos. A seguir, é apresentado o grafo gerador de caminhos restritos em recurso. Em seguida, o modelo PLIM será detalhado.

3.1. Grafo gerador de caminhos restritos em recurso

O modelo proposto utiliza uma variável para cada viagem $k = 1, ..., D$. Cada variável é gerada sob demanda através da resolução de um problema do caminho mais curto com restrição de recurso sob um grafo, que é definido formalmente nesta Seção.

Seja k um índice de viagem ($k = 1, ..., D$) e $G^k = (V^k, A^k)$ um grafo direcionado. O conjunto de vértices é dado por $V^k = \{v^k_{so}, v^k_1, v^k_2, ..., v^k_{m+n}, \bar{v}^k_1, ..., \bar{v}^k_m, v^k_{si}\}$, onde v^k_{so} e v^k_{si} são vértices *dummy*, $\{v^k_1, ..., v^k_m\}$ representam os hotéis, $\{v^k_{m+1}, ..., v^k_{m+n}\}$ são os clientes e $\bar{v}^k_h$ é uma cópia do hotel v^k_h . O conjunto de arcos é definido por $A^k = A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5$, onde $A_1 = \{(v^k_{so}, v^k_h), (\bar{v}^k_h, v^k_{si}) : h \in H\}$, $A_2 = \{(v^k_h, v^k_j) : h \in H, j \in V^+, t_{\{hj\}} \leq T_k\}$, $A_3 = \{(v^k_h, \bar{v}^k_{h'}) : h, h' \in H, h \neq h', t_{\{hh'\}} \leq T_k\}$, $A_4 = \{(v^k_i, v^k_j), (v^k_j, v^k_i) : i, j \in V^+, i < j, t_{\{ij\}} \leq T_k\}$ e $A_5 = \{(v^k_j, \bar{v}^k_h) : h \in H, j \in V^+, t_{\{hj\}} \leq T_k\}$.

Os caminhos em G^k devem começar e terminar, respectivamente, em v^k_{so} e v^k_{si}. Cada caminho p^k é associado a um recurso acumulado S_{p^k}, que é consumido ao longo da rota. No início do caminho (nó v^k_{so}), tem-se $S_{p^k} = 0$. Sempre que um arco $a \in p^k$ é percorrido, um consumo não negativo q^k_a é somado ao S_{p^k}, onde $q^k_a = 0$ se $a \in A_1$ e $q^k_{(i,j)} = t_{(i,j)}, (i, j) \in A \setminus A_1$. O intervalo de consumo acumulado permitido em cada vértice é dado por $[l^k_i, u^k_i] = [0, T_k], i \in V^k$. Se o caminho p^k começa e termina nos vértices v^k_{so} e v^k_{si}, respectivamente, sem exceder o intervalo $[l^k_i, u^k_i]$ de cada vértice visitado, dizemos que p^k é um caminho viável sobre o grafo G^k. Além disso, definimos P^k como sendo o conjunto de caminhos viáveis de G^k.

A Figura 2 define o grafo gerador de caminhos para a primeira viagem ($k = 1$) no exemplo proposto, onde o consumo de recursos é representado por q. Omitimos o grafo da segunda viagem, uma vez que ele é semelhante ao mostrado na Figura 2, mudando apenas o índice sobrescrito nos vértices para 2.

Considerando $k = 1$ e lembrando que $T_1 = 2$, os seguintes caminhos viáveis pertencem a P^1:

$p^1_1 = (v^1_{so}, v^1_1, v^1_4, \bar{v}^1_1, v^1_{si})$, $p^1_2 = (v^1_{so}, v^1_1, v^1_4, \bar{v}^1_2, v^1_{si})$, $p^1_3 = (v^1_{so}, v^1_1, \bar{v}^1_2, v^1_{si})$, $p^1_4 = (v^1_{so}, v^1_2, \bar{v}^1_1, v^1_{si})$, $p^1_5 = (v^1_{so}, v^1_2, v^1_4, \bar{v}^1_1, v^1_{si})$, $p^1_6 = (v^1_{so}, v^1_2, v^1_4, \bar{v}^1_2, v^1_{si})$, $p^1_7 = (v^1_{so}, v^1_2, \bar{v}^1_3, v^1_{si})$, $p^1_8 = (v^1_{so}, v^1_2, v^1_5, \bar{v}^1_2, v^1_{si})$, $p^1_9 = (v^1_{so}, v^1_2, v^1_5, \bar{v}^1_3, v^1_{si})$, $p^1_{10} = (v^1_{so}, v^1_3, \bar{v}^1_2, v^1_{si})$, $p^1_{11} = (v^1_{so}, v^1_3, v^1_5, \bar{v}^1_2, v^1_{si})$ e $p^1_{12} = (v^1_{so}, v^1_3, v^1_5, \bar{v}^1_3, v^1_{si})$.

Para a segunda viagem ($k = 2$), uma vez que $T_1 = T_2 = 2$, temos $P^1 = P^2$. A ideia por trás do grafo G^k é que cada caminho represente uma viagem cujo tempo correspondente não exceda o limite T_k. Para o exemplo proposto, a Tabela 1 mostra a correspondência entre os 12 caminhos pertencentes a P^1 e as viagens realizadas por eles. Por exemplo, p^1_1 corresponde a uma viagem

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

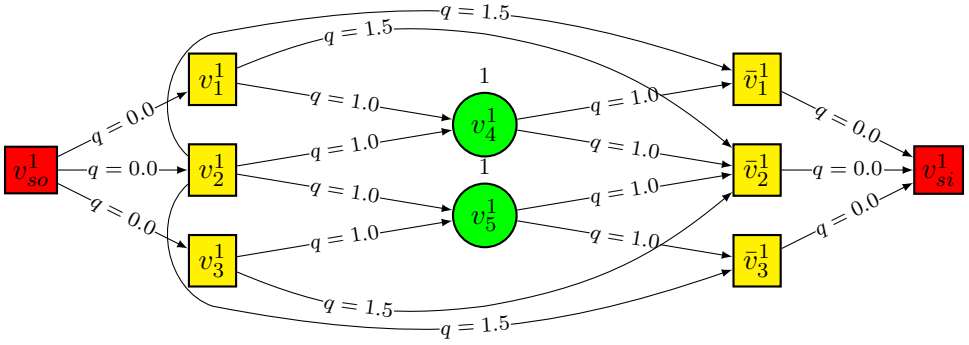


Figura 2: Grafo gerador de caminhos para a primeira viagem do exemplo ilustrativo.

que começa no Hotel 1, visita o Cliente 4 e termina no Hotel 1. Note que o caminho p^1_1 representa uma viagem restrita em recursos, mas não é utilizada em nenhuma das jornadas listadas na Seção 2. Se p^1_1 fosse usado, seria impossível concluir a segunda viagem no Hotel 3 (hotel de término de qualquer jornada), dado que $T_2 = 2$. A concatenação das viagens, bem como a imposição de que a primeira viagem começa no hotel 1 e a última termiona no hotel $|H|$, de modo a obter-se uma jornada viável, é garantida pelo modelo detalhado na Seção 3.2.

Caminho Viável	Viagem	Caminho Viável	Viagem
$p^1_1 = (v^1_{so}, v^1_1, v^1_4, \bar{v}^1_1, v^1_{si})$	(1,4,1)	$p^1_7 = (v^1_{so}, v^1_2, \bar{v}^1_3, v^1_{si})$	(2,3)
$p^1_2 = (v^1_{so}, v^1_1, v^1_4, \bar{v}^1_2, v^1_{si})$	(1,4,2)	$p^1_8 = (v^1_{so}, v^1_2, v^1_5, \bar{v}^1_2, v^1_{si})$	(2,5,2)
$p^1_3 = (v^1_{so}, v^1_1, \bar{v}^1_2, v^1_{si})$	(1,2)	$p^1_9 = (v^1_{so}, v^1_2, v^1_5, \bar{v}^1_3, v^1_{si})$	(2,5,3)
$p^1_4 = (v^1_{so}, v^1_2, \bar{v}^1_1, v^1_{si})$	(2,1)	$p^1_{10} = (v^1_{so}, v^1_3, \bar{v}^1_2, v^1_{si})$	(3,2)
$p^1_5 = (v^1_{so}, v^1_2, v^1_4, \bar{v}^1_1, v^1_{si})$	(2,4,1)	$p^1_{11} = (v^1_{so}, v^1_3, v^1_5, \bar{v}^1_2, v^1_{si})$	(3,5,2)
$p^1_6 = (v^1_{so}, v^1_2, v^1_4, \bar{v}^1_2, v^1_{si})$	(2,4,2)	$p^1_{12} = (v^1_{so}, v^1_3, v^1_5, \bar{v}^1_3, v^1_{si})$	(3,5,3)

Tabela 1: Correspondência entre caminhos e viagens para o exemplo proposto.

3.2. Formulação PLIM

Seja $\delta(j)$ o conjunto das arestas incidentes ao vértice j e α^p_a o número de vezes que o arco a é utilizado no caminho $p \in P^k$. A formulação considera as seguintes variáveis. Para cada aresta $e \in E$, a variável inteira x_e indica o número de vezes que a aresta e é usada na jornada ótima. Seja $x_e, e \in E$ uma variável inteira que indica o número de vezes que a aresta e é usada na solução. Para cada cliente $j \in V^+$, a variável binária g_j indica se o cliente $j \in V^+$ é visitado ($g_j = 1$) ou não ($g_j = 0$). Para cada viagem k e hotel h, a variável binária y^k_h indica se a viagem k começa no hotel h (y^k_h = 1) ou não (y^k_h = 0). Para cada viagem k e hotel h, a variável binária z^k_h indica se a viagem k termina no hotel h (z^k_h = 1) ou não (z^k_h = 0). Finalmente, as variáveis inteiras $\lambda^k_p, k = 1, 2, ..., D, p \in P^k$ definem o número de vezes que um caminho p é percorrido.

A formulação utiliza ainda os chamados conjuntos de mapeamento $M(x_e)$, $M(y^k_h)$ e $M(z^k_h)$ responsáveis por associar, respectivamente, as variáveis x, y, e z a um subconjunto de arcos dos grafos $G^k, k = 1, 2, ..., D$. Desse modo, definimos: $M(x_e) = \{(v^k_h, \bar{v}^k_{h'}) : k \in 1, ..., D\}, e = (h, h'), h < h', h, h' \in H$; $M(x_e) = \{(v^k_i, v^k_j), (v^k_j, v^k_i) : k = 1, ..., D\}, e = (i, j), i < j, i, j \in V^+$; $M(x_e) = \{(v^k_h, v^k_j), (v^k_j, \bar{v}^k_h) : k = 1, ..., D\}, e = (h, j), h \in H, j \in V^+$;

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

$M(y_h^k) = \{(v_{so}^k, v_h^k)\}, h \in H, k = 1, ..., D$; e $M(z_h^k) = \{(v_h^k, v_{si}^k)\}, h \in H, k = 1, ..., D$. O modelo (1) apresenta a formulação PLIM para o OPHS.

$$\text{Max} \sum_{j \in V^+} s_j g_j \tag{1a}$$

$$\text{S.t.} \sum_{e \in \delta(j)} x_e = 2g_j, \qquad j \in V^+; \tag{1b}$$

$$\sum_{e \in E} t_e x_e \leq T_{max}; \tag{1c}$$

$$y_1^1 = 1; \tag{1d}$$

$$z_m^D = 1; \tag{1e}$$

$$y_h^k = z_h^{k-1} \qquad h \in H, k = 2, ...D; \tag{1f}$$

$$x_e = \sum_{k=1}^{D} \sum_{p \in P^k} \left(\sum_{a \in M(x_e)} \alpha_a^p \right) \lambda_p^k \qquad \forall e \in E; \tag{1g}$$

$$y_h^k = \sum_{p \in P^k} \left(\sum_{a \in M(y_h^k)} \alpha_a^p \right) \lambda_p^k \qquad \forall k = 1, ..., D, h \in H; \tag{1h}$$

$$z_h^k = \sum_{p \in P^k} \left(\sum_{a \in M(z_h^k)} \alpha_a^p \right) \lambda_p^k \qquad \forall k = 1, ..., D, h \in H; \tag{1i}$$

$$\sum_{p \in P^k} \lambda_p^k = 1 \qquad \forall k = 1, ..., D; \tag{1j}$$

$$x_e \in \mathbb{Z} \qquad \forall e \in E; \tag{1k}$$

$$\lambda_p^k \in \mathbb{Z} \qquad \forall k = 1, ..., D, p \in P^k; \tag{1l}$$

$$g_j \in \{0, 1\} \qquad \forall j \in V^+ \tag{1m}$$

$$y_h^k \in \{0, 1\} \qquad \forall k = 1, ..., D, h \in H; \tag{1n}$$

$$z_h^k \in \{0, 1\} \qquad \forall k = 1, ..., D, h \in H \tag{1o}$$

No modelo (1), (1a) define a função objetivo, maximizando a pontuação total obtida na jornada. As Restrições (1b) garantem que se um cliente é visitado, a soma das arestas correspondentes incidentes neste cliente é exatamente 2. Em outras palavras, cada cliente é visitado no máximo uma vez. A Restrição (1c) diz respeito ao tempo máximo da jornada, que não pode exceder T_{max}. A seguir, a Restrição (1d) define que o hotel 1 inicia a primeira viagem, e (1e) garante que a última viagem termina no hotel m. As Restrições (1f) garantem que as viagens são corretamente concatenadas para construir uma jornada. Nas Restrições (1g), (1h) e (1i), respectivamente, as variáveis x_e, y_j^k e z_j^k são escritas em função das variáveis λ. A ideia é que os valores de x_e, y_j^k ou z_j^k sejam definidos pela soma do número de vezes que algum arco mapeado nessas variáveis (i.e. pertencentes aos conjuntos $M(x_e)$, $M(y_h^k)$ e/ou $M(z_h^k)$) é usado pelos caminhos $p \in P^k$ com valor λ_p^k. Em outras palavras, os conjuntos de Restrições (1g), (1h), (1i) e (1l) definem a conexão entre os caminhos $p \in P^k$, obtidos sobre o grafo gerador G^k, com o restante da formulação. As Restrições (1j)

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

são usadas para garantir que um caminho seja escolhido para cada viagem. As demais restrições definem o domínio das variáveis.

A Figura 3 mostra os grafos geradores de caminhos para o exemplo ilustrativo, onde em cada arco a é colocada uma variável se a pertence ao conjunto de mapeamento dessa variável. Seja por exemplo, a variável $x_{(1,4)}$. De acordo com a definição, tem-se $M(x_{(1,4)}) = \{(v_1^1, v_4^1), (v_4^1, \bar{v}_1^1), (v_1^2, v_4^2), (v_4^2, \bar{v}_1^2)\}$. Logo, essa variável está indicada sobre cada um dos arcos em $M(x_{\{1,4\}})$.

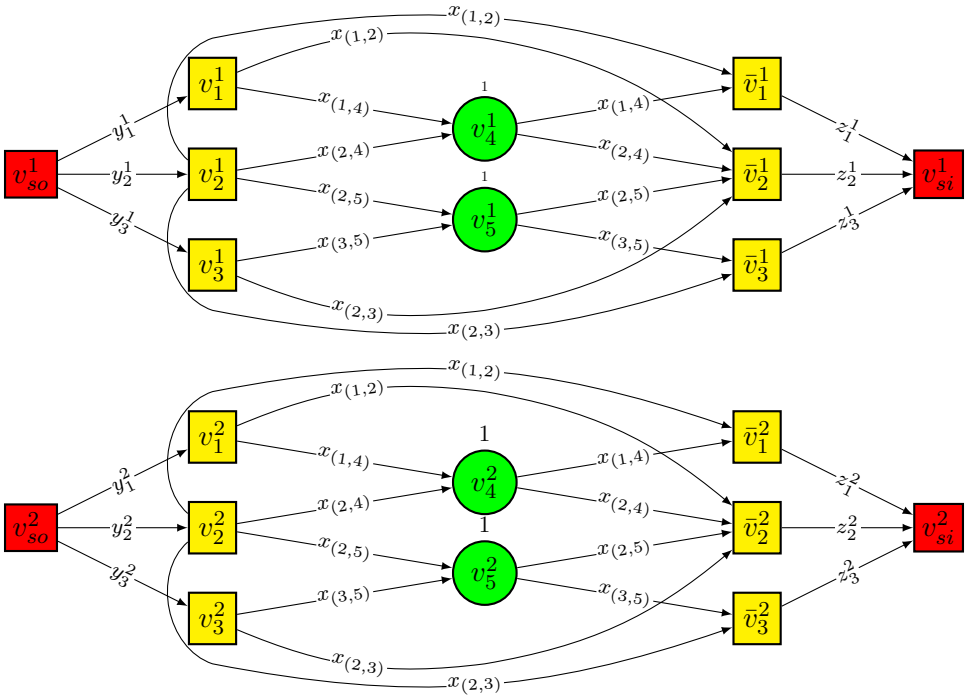


Figura 3: Variáveis mapeadas nos arcos dos grafos geradores de caminho para a instância ilustrativa.

Agora, a Formulação (1) aplicada à instância proposta na Seção 2 é apresentada. As variáveis x associadas a arestas com custo maior do que T_{max} e o domínio das variáveis foram omitidos. A formulação segue.

$$
\begin{aligned}
&\text{Max } g_4 + g_5 \\
&\text{S.a. } x_{(1,4)} + x_{(2,4)} = 2g_4; \\
&\quad x_{(2,5)} + x_{(3,5)} = 2g_5; \\
&\quad 1.5x_{(1,2)} + x_{(1,4)} + 1.5x_{(2,3)} + x_{(2,5)} + x_{(3,5)} \leq 4; \\
&\quad y_1^1 = 1; \\
&\quad z_3^2 = 1; \\
&\quad y_1^2 = z_1^1; \\
&\quad y_2^2 = z_2^1; \\
&\quad y_3^2 = z_3^1;
\end{aligned}
$$

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

$$
\begin{aligned}
&y_1^1 + y_2^1 + y_3^1 = 1;\\
&y_1^2 + y_2^2 + y_3^2 = 1;\\
&z_1^1 + z_2^1 + z_3^1 = 1;\\
&z_1^2 + z_2^2 + z_3^2 = 1;\\
&\lambda_{p_1}^1 + ... + \lambda_{p_{12}}^1 = 1;\\
&\lambda_{p_1}^2 + ... + \lambda_{p_{12}}^2 = 1;\\
&x_{(1,2)} = \lambda_{p_3^1}^1 + \lambda_{p_4^1}^1 + \lambda_{p_3^2}^2 + \lambda_{p_4^2}^2;\\
&x_{(1,4)} = 2\lambda_{p_1^1}^1 + \lambda_{p_2^1}^1 + \lambda_{p_5^1}^1 + 2\lambda_{p_1^2}^2 + \lambda_{p_2^2}^2 + \lambda_{p_5^2}^2;\\
&x_{(2,3)} = \lambda_{p_7^1}^1 + \lambda_{p_{10}^1}^1 + \lambda_{p_7^2}^2 + \lambda_{p_{10}^2}^2;\\
&x_{(2,4)} = \lambda_{p_2^1}^1 + \lambda_{p_5^1}^1 + 2\lambda_{p_6^1}^1 + \lambda_{p_2^2}^2 + \lambda_{p_5^2}^2 + \lambda_{p_6^2}^2;\\
&x_{(2,5)} = 2\lambda_{p_8^1}^1 + \lambda_{p_9^1}^1 + \lambda_{p_{11}^1}^1 + 2\lambda_{p_8^2}^2 + \lambda_{p_9^2}^2 + \lambda_{p_{11}^2}^2;\\
&x_{(3,5)} = \lambda_{p_9^1}^1 + \lambda_{p_{11}^1}^1 + 2\lambda_{p_{12}^1}^1 + \lambda_{p_9^2}^2 + \lambda_{p_{11}^2}^2 + \lambda_{p_{12}^2}^2;\\
&y_1^1 = \lambda_{p_1^1}^1 + \lambda_{p_2^1}^1 + \lambda_{p_3^1}^1;\\
&y_1^2 = \lambda_{p_1^2}^2 + \lambda_{p_2^2}^2 + \lambda_{p_3^2}^2;\\
&y_2^1 = \lambda_{p_4^1}^1 + \lambda_{p_5^1}^1 + \lambda_{p_6^1}^1 + \lambda_{p_7^1}^1 + \lambda_{p_8^1}^1 + \lambda_{p_9^1}^1;\\
&y_2^2 = \lambda_{p_4^2}^2 + \lambda_{p_5^2}^2 + \lambda_{p_6^2}^2 + \lambda_{p_7^2}^2 + \lambda_{p_8^2}^2 + \lambda_{p_9^2}^2;\\
&y_3^1 = \lambda_{p_{10}^1}^1 + \lambda_{p_{11}^1}^1 + \lambda_{p_{12}^1}^1;\\
&y_3^2 = \lambda_{p_{10}^2}^2 + \lambda_{p_{11}^2}^2 + \lambda_{p_{12}^2}^2;\\
&z_1^1 = \lambda_{p_1^1}^1 + \lambda_{p_4^1}^1 + \lambda_{p_5^1}^1;\\
&z_1^2 = \lambda_{p_1^2}^2 + \lambda_{p_4^2}^2 + \lambda_{p_5^2}^2;\\
&z_2^1 = \lambda_{p_2^1}^1 + \lambda_{p_3^1}^1 + \lambda_{p_6^1}^1 + \lambda_{p_8^1}^1 + \lambda_{p_{10}^1}^1 + \lambda_{p_{11}^1}^1;\\
&z_2^2 = \lambda_{p_2^2}^2 + \lambda_{p_3^2}^2 + \lambda_{p_6^2}^2 + \lambda_{p_8^2}^2 + \lambda_{p_{10}^2}^2 + \lambda_{p_{11}^2}^2;\\
&z_3^1 = \lambda_{p_7^1}^1 + \lambda_{p_9^1}^1 + \lambda_{p_{12}^1}^1;\\
&z_3^2 = \lambda_{p_7^2}^2 + \lambda_{p_9^2}^2 + \lambda_{p_{12}^2}^2;\\
&x \in \mathbb{Z}; \lambda \in \mathbb{Z}; g \in \{0,1\}; y \in \{0,1\}; z \in \{0,1\}.
\end{aligned}
$$

3.3. Resolvendo a formulação PLIM através da ferramenta VRPSolver

A fim de usarmos o VRPSolver, fornecemos à ferramenta os grafos geradores de caminhos e a formulação definidos anteriormente. Para cada grafo, deve-se informar se o recurso que está sendo é o ou não do tipo *main*. De acordo com Pessoa et al. [2020], um recurso do tipo *main* deve ser monótono, o que é o caso dos recursos utilizados nesse trabalho. Já quando fornecemos a formulação, as Restrições (1b)-(1f), (1k), (1m)-(1o) são inseridas no problema mestre. Por outro lado, as Restrições (1g)–(1j) e (1l) são definidas automaticamente. Mais especificamente, (1g)–(1i) são derivadas do mapeamento (i.e. $M(x_e), M(y_h^k)$, e $M(z_h^k)$) e (1j) é definida a partir do fornecimento do número mínimo e máximo de caminhos que devem ser obtidos em cada grafo G^k em uma solução. No nosso caso, fornecemos tanto o mínimo quanto o máximo iguais a 1, já que

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

queremos que uma viagem seja definida em cada grafo.

Além dos grafos e da formulação PLIM, fornecemos ainda mais duas informações à ferramenta: a primeira é que o *branch* pode ser feito em todas as variáveis inteiras g, x, y e z com o mesmo grau de prioridade e a segunda informação é um conjunto de parâmetros relativos ao algoritmo BCP usado pela ferramenta. A respeito destes parâmetros, fornecemos os mesmos usados por Pessoa et al. [2020] para o clássico problema de roteamento de veículos capacitado.

O VRPSolver permite ainda ao usuário definir uma coleção $\mathcal{P}^V$ dos chamados *packing sets* associados ao modelo. Um *packing set* é definido como um subconjunto de vértices dos grafos geradores de recursos onde no máximo um dos vértices desse subconjunto pode ser visitado por um dos caminhos e no máximo uma vez. Uma vez definidos, o VRPSolver habilita técnicas estado da arte de algoritmos BCP para problemas de roteamento, como por exemplo, *branching* e enumeração de caminhos. A definição de *packing sets* apropriados para um dado problema é uma tarefa de modelagem do usuário. Para mais detalhes, veja [Pessoa et al., 2020]. No OPHS, por definição, qualquer cliente é visitado no máximo uma vez. Assim, no modelo proposto, é definido, para cada cliente $i \in V^+$ um *packing set* $\mathcal{P}_i$ composto por todas as representações de i nos grafos geradores de caminhos. Matematicamente, $\mathcal{P}_i = \{v_i^k : k = 1, \ldots, D\}$ e $\mathcal{P}^V = \cup_{i \in V_+} \{\mathcal{P}_i\}$.

Após o fornecimento de todas as informações prévias, o VRPSolver usa as restrições (1b), (1g), (1h) e (1i) para reescrever a Formulação (1) apenas em função das variáveis λ. Essa nova formulação é resolvida através de um algoritmo BCP genérico onde as variáveis λ são geradas sob demanda através da resolução de problemas do caminho mais curto com restrição de recurso sobre os grafos G^k. A formulação é resolvida em cada nó da árvore de *branch-and-bound*. *Branching* nas variáveis inteiras g, x, y e z é usado para resolver o problema até a otimalidade.

4. Experimento Computacionais

Nesta seção, apresenta-se os resultados computacionais que demonstram a robustez do modelo VRPSolver proposto para o OPHS. O modelo foi testado nas instâncias de *benchmark* propostas por Divsalar et al. [2013]. Esses autores geraram as instâncias a partir daquelas do OP, alterando o número de hotéis $|H|$ e o número de viagens D, de modo que o valor da solução ótima para cada instância OPHS seja o mesmo da instância OP correspondente. As instâncias podem ser acessadas em `https://www.mech.kuleuven.be/en/cib/op` e para mais detalhes de como foram geradas, consulte Divsalar et al. [2013].

Todos os experimentos foram conduzidos em *single thread* de um computador com processador Intel Core i7-10700 com 2.90 GHz e 16 GB de RAM e sistema operacional Ubuntu 18.04.2 LTS. O modelo foi implementado em linguagem de programação Julia com a versão v0.4.1 do VRPSolver, esse, por sua vez, utilizou o CPLEX v12.10 como resolvedor PL e PLIM. O tempo máximo de testes de cada instância foi limitado em 10800 segundos. Além disso, para cada uma delas, foi fornecido ao modelo um limite inferior viável da função objetivo obtido a partir de Sohrabi et al. [2021].

4.1. Resultados

A Tabela 2 umariza os resultados obtidos. A coluna *Nome* exibe o nome da instância OP usada para gerar as instâncias OPHS correspondentes. Cada nome consiste em dois termos separados por um hífen. As instâncias que começam com "T1" e "T3" têm 32 e 33 nós, respectivamente. No caso das instâncias que começam com um valor numérico, esse corresponde ao número total de nós. O segundo termo indica o tempo máximo da jornada completa (T_{max}). A coluna *Opt* denota o valor da solução ótima, que é o mesmo para todas as instâncias na mesma linha. As colunas restantes indicam o tempo computacional em segundos para cada par $(|H|, |D|)$ testamos. Neste trabalho, nos concentramos especificamente em instâncias com $|H| \leq 8$. O termo "TL" aparece na coluna

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

		Tempo(s)							Tempo(s)				
Nome	opt	(3,2)	(4,3)	(5,4)	(7,3)	(8,4)	Name	opt	(3,2)	(4,3)	(5,4)	(7,3)	(8,4)
T1-65	240	1,0	0,9	0,9	1,1	5,4	64-65	1116	11,0	4,8	7,5	ML	346,9
T1-70	260	1,1	1,0	1,0	3,2	1,4	64-70	1188	23,7	3,2	148,9	3337,2	1001,9
T1-73	265	1,0	1,0	1,0	4,2	19,4	64-75	1236	TL	91,5	93,7	ML	TL
T1-75	270	1,1	1,0	1,0	1,1	3,6	64-80	1284	4069,4	42,5	17,6	1647,0	ML
T1-80	280	1,3	1,2	1,0	1,2	4,2	66-40	575	1,0	1,0	1,1	1,2	1,2
T1-85	285	1,5	1,1	1,1	4,7	3,8	66-45	650	1,1	1,1	1,1	1,6	1,7
T3-65	610	1,2	1,1	1,0	34,0	6,8	66-50	730	1,1	1,0	1,1	1,4	1,5
T3-75	670	1,3	1,1	1,0	47,3	1,1	66-55	825	1,1	1,1	1,3	1,7	1,4
T3-80	710	1,4	1,1	1,1	39,4	12,0	66-60	915	1,2	1,2	1,2	2,6	3,7
T3-85	740	1,3	1,2	1,1	1,5	15,4	66-125	1670	12060,6	511,1	21,0	ML	TL
T3-90	770	1,8	1,4	1,2	2,8	49,4	66-130	1680	3426,3	ML	85,5	ML	ML
T3-95	790	1,7	1,4	1,2	14,1	44,0	100-30	173	1,2	1,3	1,5	1,4	1,5
T3-100	800	2,3	8,5	1,2	611,4	235,4	100-35	241	1,3	1,3	1,5	1,4	1,4
T3-105	800	7,6	20,7	3,8	1257,5	8,5	100-40	299	1,4	1,3	1,5	1,3	1,5
64-45	816	1,2	1,4	1,2	22,4	2,3	100-45	367	1,5	1,3	1,5	1,4	1,5
64-50	900	1,9	1,6	1,8	12,7	10,8	102-50	181	1,2	1,2	1,4	1,3	1,5
64-55	984	1,9	2,5	1,8	31,4	57,1	102-60	243	1,2	1,3	1,4	1,3	1,4
64-60	1062	2,5	2,5	2,0	405,0	23,8							
Média		1,8	2,8	1,3	138,6	28,0			1225,3	41,6	22,9	384,7	105,2

Tabela 2: Sumário dos resultados.

Tempo(s) quando a resolução dessa instância atingiu o limite de tempo especificado, não sendo encontrado o ótimo. Algumas instâncias produziram um erro de memória devido ao grande tamanho do subproblema de *pricing*, sendo estas registradas com o termo "ML" na coluna *Tempo(s)*.

Ao observar a Tabela 2, notamos que o algoritmo proposto conseguiu resolver de forma ótima 165 de 175 instâncias dentro do limite de tempo. Algumas instâncias tinham um pequeno número de nós, mas não foram resolvidas de forma ótima, como a instância 64-65 com $|H| = 7$ e $D = 3$. Uma das razões que contribuem para isso é o aumento do número de hotéis, que aumenta o número de caminhos possíveis. Outro fator é o número médio de clientes visitados em cada viagem (16,7), que tende a crescer com a diminuição do número de viagens permitidas. Ambos os fatores (tamanho do conjunto P^k e tamanho médio dos caminhos) contribuem para a dificuldade de resolver os subproblemas de *pricing*.

4.2. Ilustração de uma solução

A Figura 4 apresenta a solução ótima encontrada para a instância 64-45 com 5 hotéis e 4 viagens. Um ponto importante da solução é que esta usa todos os hotéis exatamente uma vez. Ømesmo acontece com todas as instâncias aqui testadas. Porém, isso se deve apenas a maneira como as instâncias foram geradas. Por definição, nada impediria de existir uma instância para o OPHS cuja solução ótima use um hotel mais de uma vez ou deixe de visitar alguns deles.

5. Conclusão

Este artigo apresentou um novo método exato para resolver o *Orienteering Problem* com seleção de hotéis, que até o momento só tinha sido abordado com métodos heurísticos. O método proposto se baseia em uma modelagem VRPSolver, um solucionador avançado que utiliza um algoritmos BCP genéricos. O estudo detalha construção do grafo gerador de caminhos restritos em recurso, além da formulação do problema a partir de um modelo de Otimização Linear Inteira Mista.

Vários experimentos computacionais foram realizados considerando instâncias previamente propostas na literatura para o problema OPHS, com o intuito de demonstrar a robustez do

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

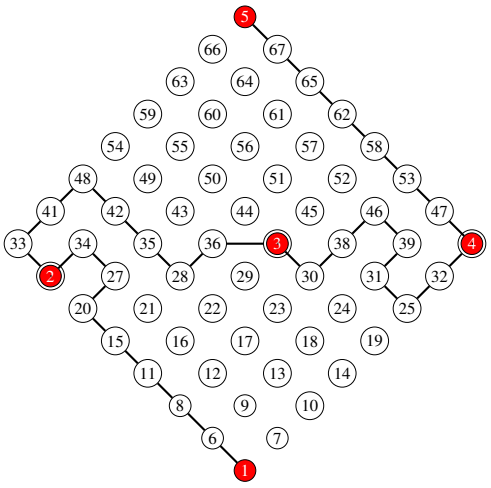


Figura 4: Solução ótima para a instância 64-45-3-4

método proposto. Foram encontradas soluções ótimas para 165 das 175 instâncias testadas, onde considerou-se instâncias com no máximo 102 nós, 8 hotéis e 4 viagens dentro do limite de tempo de execução especificado. Os resultados obtidos são promissores e podem ser usados como referência em futuros estudos sobre o OPHS, bem como em aplicações práticas que envolvam, por exemplo, o planejamento de rotas turísticas e de entregas.

Para futuras pesquisas, planeja-se estender o modelo proposto para outros problemas de seleção de hotéis VRP, como o Problema do Caixeiro Viajante com Seleção de Hotéis (TSPHS) [Vansteenwegen et al., 2012]. A principal distinção entre o OPHS e o TSPHS consiste em que, no TSPHS, é necessário visitar todos os clientes e a função objetivo visa minimizar a distância total percorrida.

Referências

Ataei, M., Divsalar, A., e Saberi, M. (2022). The bi-objective orienteering problem with hotel selection: an integrated text mining optimisation approach. *Information Technology and Management*. ISSN 1385-951X. URL https://link.springer.com/10.1007/s10799-022-00377-5.

Ben Mohamed, I., Klibi, W., Sadykov, R., Şen, H., e Vanderbeck, F. (2023). The two-echelon stochastic multi-period capacitated location-routing problem. *European Journal of Operational Research*, 306(2):645–667. ISSN 03772217. URL https://linkinghub.elsevier.com/retrieve/pii/S0377221722005756.

Clementino, T., Rosas, J., De Freitas, R., e Uchoa, E. (2022). Solving real urban VRPTW instances by applying a Branch-Cut-and-Price via VRPsolver. In *2022 XVLIII Latin American Computer Conference (CLEI)*, p. 1–8. IEEE. ISBN 978-1-6654-7671-3. URL https://ieeexplore.ieee.org/document/9959966/.

Costa, L., Contardo, C., e Desaulniers, G. (2019). Exact Branch-Price-and-Cut Algorithms for Vehicle Routing. *Transportation Science*, 53(4):946–985. ISSN 0041-1655. URL https://pubsonline.informs.org/doi/10.1287/trsc.2018.0878.

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914

Divsalar, A., Vansteenwegen, P., e Cattrysse, D. (2013). A variable neighborhood search method for the orienteering problem with hotel selection. *International Journal of Production Economics*, 145(1):150–160.

Divsalar, A., Vansteenwegen, P., Sörensen, K., e Cattrysse, D. (2014). A memetic algorithm for the orienteering problem with hotel selection. *European Journal of Operational Research*, 237(1): 29–49.

Freitas, M., Silva, J. M. P., e Uchoa, E. (2023). A unified exact approach for Clustered and Generalized Vehicle Routing Problems. *Computers & Operations Research*, 149: 106040. ISSN 03050548. URL https://linkinghub.elsevier.com/retrieve/pii/S0305054822002702.

Gunawan, A., Lau, H. C., e Vansteenwegen, P. (2016). Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255 (2):315–332.

Pessoa, A., Sadykov, R., e Uchoa, E. (2021). Solving Bin Packing Problems Using VRPSolver Models. *Operations Research Forum*, 2(2):20. ISSN 2662-2556. URL https://link.springer.com/10.1007/s43069-020-00047-8.

Pessoa, A., Sadykov, R., Uchoa, E., e Vanderbeck, F. (2020). A generic exact solver for vehicle routing and related problems. *Mathematical Programming*, 183(1):483–523.

Queiroga, E., Sadykov, R., e Uchoa, E. (2021). A POPMUSIC matheuristic for the capacitated vehicle routing problem. *Computers & Operations Research*, 136:105475. ISSN 03050548. URL https://linkinghub.elsevier.com/retrieve/pii/S0305054821002240.

Roboredo, M., Sadykov, R., e Uchoa, E. (2023). Solving vehicle routing problems with intermediate stops using VRPSolver models. *Networks*, 81(3):399–416. ISSN 0028-3045. URL https://onlinelibrary.wiley.com/doi/10.1002/net.22137.

Sohrabi, S., Ziarati, K., e Keshtkaran, M. (2020). A greedy randomized adaptive search procedure for the orienteering problem with hotel selection. *European Journal of Operational Research*, 283(2):426–440.

Sohrabi, S., Ziarati, K., e Keshtkaran, M. (2021). Acs-ophs: Ant colony system for the orienteering problem with hotel selection. *EURO Journal on Transportation and Logistics*, 10:100036.

Toledo, A., Riff, M.-C., e Neveu, B. (2019). A hyper-heuristic for the orienteering problem with hotel selection. *IEEE Access*, 8:1303–1313.

Vansteenwegen, P., Souffriau, W., e Sörensen, K. (2012). The travelling salesperson problem with hotel selection. *Journal of the Operational Research Society*, 63(2):207–217.

Vansteenwegen, P., Souffriau, W., e Van Oudheusden, D. (2011). The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10.

DOI: 10.59254/sbpo-2023-174914
http://live-proceedings.pantheonsite.io/p/174914