UMA META-HEURÍSTICA GREEDY RANDOMIZED ADAPTIVE SEARCH PROCEDURES (GRASP) PARA O PROBLEMA P-HUB DE MÁXIMA COBERTURA NÃO CAPACITADO COM ALOCAÇÃO ÚNICA

Warley Ferreira da Cunha

Centro Federal de Educacão Tecnológica de Minas Gerais - CEFET-MG.

Av. Amazonas, 7.675, Bairro Nova Gameleira, Belo Horizonte/MG.

warleycunha@yahoo.com.br

ABSTRACT. This work describes a Greedy Randomized Adaptive Search Procedures (GRASP) meta-heuristic for the uncapacitated maximum coverage p-hub problem with single allocation. The objective is to determine the best location for p-hubs and the assignment of each one of the non-hub nodes to a single hub, so that the total demand between pairs of nodes within of a given coverage distance is maximized. We consider all nodes as possible candidates for establishing hub facilities, which increases the complexity of the problem. Computational tests are applied to compare the objective function value with respect to CPLEX. The results were not satisfactory, therefore adjustments to the algorithm will be made in order to find good solutions in better computational time than CPLEX

Keywords: Hub Location, Meta-Heuristics, Greedy Randomized Adaptive Search Procedures.

RESUMO. Este trabalho descreve uma meta-heurística Greedy Randomized Adaptive Search Procedures (GRASP) para o problema p-hub de máxima cobertura não capacitado com alocação única. O objetivo é determinar a melhor localização para p-hubs e a atribuição de cada um dos nós não hubs a um único hub, de modo que a demanda total entre pares de nós dentro de uma determinada distância de cobertura seja maximizada. Consideramos todos os nós como possíveis candidatos ao estabelecimento de instalações de hub, o que aumenta a complexidade do problema. Testes computacionais são aplicados para comparação do valor de função objetivo com relação ao CPLEX. Os resultados não foram satisfatórios, portanto ajustes no algoritmo serão realizados afim de encontrar boas soluções em tempo computacional melhor que o CPLEX.

Palavras Chave: Localização de hubs, Meta-Heurística, Greedy Randomized Adaptive Search Procedures.

1 Introdução

Arquiteturas *hub-and-spoke (HS)* são frequentemente usadas em projetos de redes de grande escala, tais como as encontradas em companhias aéreas de passageiros, serviços postais, telecomunicações e sistemas de trânsito rápido. Nessas estruturas de redes, com fluxos entre muitos pares de origem-destino $O-D$, mercadorias de diferentes origens são enviadas para instalações intermediárias, conhecidas como hubs, que são responsáveis pela agregação e distribuição do fluxo para vários destinos. Os hubs, são instalações especiais que servem como pontos de comutação, transbordo e classificação em sistemas de distribuição, permitem a conexão de um grande número de nós com um pequeno número de arcos entre-hubs, reduzindo a infraestrutura e o custo operacional (O'Kelly e Miller 1994).

Outra vantagem importante de uma rede *hub-and-spoke*, também denominada rede *eixo-raio*, é que as instalações dos hubs podem ser conectadas por caminhos altamente eficientes, permitindo alcançar economias de escala nos custos de transporte (ou tempo de viagem) entre os hubs.

Definir hubs e alocar nós não hub (spokes) para nós hub de modo que seja otimizado o transporte entre os pontos de origem e destino é um problema denominado Problema de Localização de Hubs (PLH). Os PLH são contidos por especificidades e deles extraem-se algumas variantes, dentre essas uma denominada problema p-hub de máxima cobertura não capacitado com alocação única (Uncapacitated Single Allocation p-Hub Maximal Covering Problem - USApHMCP) que consiste em determinar p nós para serem hubs entre um conjunto de nós candidatos e alocar cada nó não hub a exatamente um único hub para maximizar a demanda total coberta entre pares de nós dentro de uma determinada distância.

Mais especificamente, neste trabalho considera-se o caso em que todos os pares de hubs estão conectados e, a quantidade de hubs, denotada por p, é conhecida. Um nó não hub pode ser alocado a somente um único nó hub (alocação única) e os arcos e os hubs não possuem limite de capacidade (não capacitado). A Figura 1 ilustra a configuração de uma rede com 25 nós, sendo 3 hubs instalados e 22 spokes, com suas respectivas alocações.

Na literatura, esse problema é classificado como NP-difícil, ou seja, o esforço computacional cresce exponencialmente à medida que aumenta o número de nós para resolver o problema. Assim, formulações de programação inteira podem apresentar dificuldade para alcançar as soluções ótimas do USApHMCP à medida que a dimensão do problema cresce. Mediante isso, o objetivo do trabalho é aplicar a meta-heurística Greedy Randomized Adaptive Search Procedures (GRASP) com busca local em vizinhança variável para sua solução. Os resultados dos experimentos computacionais utilizando o algoritmo proposto serão comparados com os resultados obtidos pelo solver CPLEX, usando as instâncias da literatura Australian Post (AP).

O trabalho está estruturado da seguinte forma: Na Seção 2, apresentamos a caracterização e o modelo matemático para o USApHMCP. A Seção 3 detalha o algoritmo proposto, enquanto os experimentos computacionais são expostos na Seção 4. Considerações finais e sugestões para aprimoramento da abordagem adotada são discutidas na Seção 5.

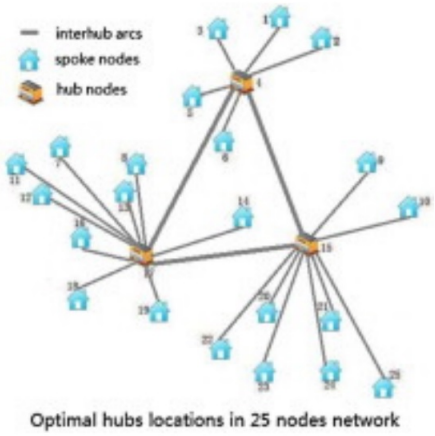

Figura 1: Extraída de Yang et al, 2013.

2 Caracterização do problema

Nesta seção, apresentamos o modelo matemático utilizado para resolver o USApHMCP. Diferentes formulações matemáticas a respeito da alocação de p-hubs podem ser encontradas na literatura. Especificamente, neste trabalho, usaremos a formulação proposta em Peker e kara (2015).

Seja N o conjunto de nós, cuja cardinalidade será denotada por $n = |N|$. Um determinado número p de nós é selecionado como hubs, cujo conjunto é dado por H, isto é, $p = |H|$, $H \subset N$. Sejam c_{ij} e W_{ij} o custo unitário de transporte e o fluxo do nó de origem i até o nó destino j, respectivamente. Assume-se que um nó não hub pode usar apenas um nó hub para se comunicar com os outros nós. O custo de transporte do nó $i \in N$, atribuído ao hub $k \in H$ para o nó $j \in N$, atribuído ao hub $m \in H$ é dado por $\gamma c_{ik} + \alpha c_{km} + \delta c_{mj}$, sendo γ, α, δ taxas unitárias para coleta (origem-hub), transferência (hub-hub) e distribuição (hub-destino), respectivamente. O parâmetro α está associado à economia de escala de transporte hub-hub, portanto, espera-se $\alpha \leq \gamma$ e $\alpha \leq \delta$.

O par (i, j) $O - D$ é coberto pelos hubs k e m se o custo de i para j via hubs k e m não excede o valor específico β_{ij} (custo de transporte que indica o nível de cobertura para o par $O - D$). Veja Figura 2.

Seja a_{ij}^{km} um parâmetro do modelo tal que:

$$a_{ij}^{km} = \begin{cases} 1, & \text{se} \quad \gamma c_{ik} + \alpha c_{km} + \delta c_{mj} \leq \beta_{ij}, \\ 0, & \text{caso contrário.} \end{cases}$$

Por fim, considere o parâmetro auxiliar $\lambda_{ij} = max_{km} a_{ij}^{km}, \forall i, j \in N$. Defina, então, a

variável binária de decisão, de modo que

$$x_{ij} = \begin{cases} 1, & \text{se o nó } i \text{ for atribuído ao hub } j, \\ 0, & \text{caso contrário.} \end{cases}$$

Além disso,

$$x_{ii} = \begin{cases} 1, & \text{se nó } i \text{ for hub,} \\ 0, & \text{caso contrário.} \end{cases}$$

A variável de decisão não negativa z_{ij} representa a fração do fluxo roteado do nó de origem i para o nó de destino j que é coberto. Mediante o descrito acima, uma vez lidos os dados de entrada, o passo inicial é calcular os parâmetros a_{ij}^{km} conforme detalhado no Algoritmo 1.

Algorithm 1 Procedimento para calcular os parâmetros de cobertura a_{ij}^{km}

input: $N, \gamma, \alpha, \delta, \beta_{ij}$
output: todos os a_{ij}^{km}

for all $(i, j, k, m) \in N$ **do**
  if $\gamma C_{ik} + \alpha C_{km} + \delta C_{mj} \leq \beta_{ij}$ **then**
    $a_{ij}^{km} = 1$
  else
    $a_{ij}^{km} = 0$
  end if
end for

O USApHMCP pode ser formulado como segue:

$$max \sum_{i \in N} \sum_{j \in N} W_{ij} z_{ij} \tag{1}$$

suj. a

$$\sum_{k \in N} x_{kk} = p \tag{2}$$

$$\sum_{k \in N} x_{ik} = 1, \quad \forall i \in N. \tag{3}$$

$$x_{ik} \leq x_{kk}, \quad \forall i, k \in N. \tag{4}$$

$$z_{ij} \leq \sum_{k \in N} a_{ij}^{km} x_{ik} + \lambda_{ij}(1 - x_{jm}), \quad \forall i, j \in N, m \in N. \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \in N. \tag{6}$$

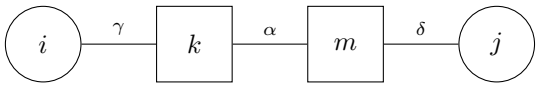

Figura 2: Rede de hubs e não hubs. Espera-se $\alpha \leq \gamma; \alpha \leq \delta$.

$$z_{ij} \geq 0, \quad \forall i, j \in N. \tag{7}$$

A função objetivo (1) visa maximizar a demanda total coberta entre todos os pares de $O - D$ (origem e destino). As restrições (2) garantem que exatamente p hubs são instalados, enquanto as restrições (3) impõem que cada nó i seja atribuído a exatamente um hub selecionado. Restrições (4) garantem que nenhum nó seja atribuído a um hub que não está aberto naquela localização. As restrições (5) determinam o limitante superior para a fração de fluxo entre os nós i e j que é coberto. Finalmente, as restrições (6) e (7) são as restrições de domínio das variaveis.

3 Descrição do Algoritmo Proposto

Esta seção descreve a metodologia utilizada para a solução do problema abordado, incluindo a representação e construção da solução, a função de avaliação, a busca local, os movimentos e a meta-heurística GRASP implementada.

3.1 Representação da solução

A solução será representada por dois vetores (s, H). O vetor $s = (s_1, ..., s_n)$ possui n elementos e contém as alocações de nó, em que s_i indica o hub ao qual o nó i está alocado. O vetor H, de tamanho p, contém os hubs selecionados, em que $H = (h_1, ..., h_p)$. A figura 3 representa uma rede de hubs e não hubs e suas alocações.

3.2 Solução inicial

A solução inicial é construída de forma gulosa elemento a elemento. Para este trabalho do USApHMCP, a construção inicial gulosa será feita escolhendo como hubs os p nós com maior fluxo (que é dado pela soma das demandas originadas e as demandas destinadas ao nó). Em seguida, é feita uma alocação inicial gulosa, alocando cada nó não hub ao hub mais próximo.

3.3 Função de avaliação

A função de avaliação avalia a qualidade de uma solução. Dada uma solução (s, H), a função de avaliação utilizada é:

$$f(s) = \sum_{i \in N} \sum_{j \in N} W_{ij} a_{ij}^{s_i s_j}.$$

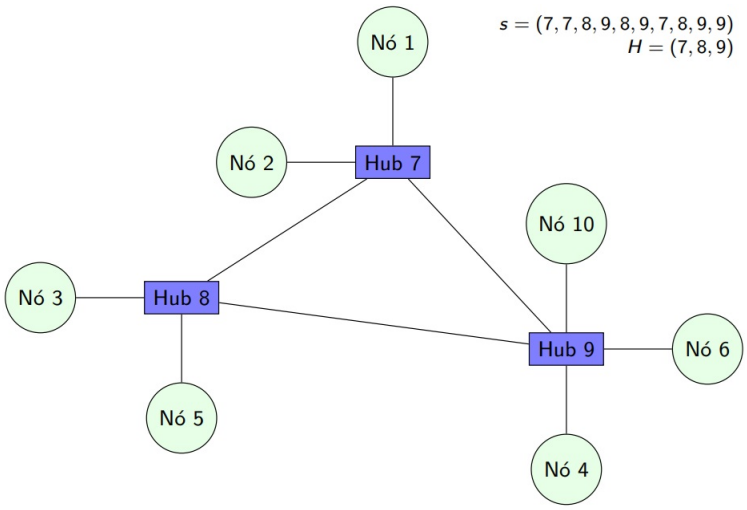

Figura 3: Representação de uma rede de hubs e não hubs e suas alocações.

3.4 Busca local

A busca local utilizada foi o método de descida em vizinhança variável *(Variable Neighborhood Descent* - VND). Proposto por Mladenovic e Hansen [1997] e explora o espaço de solucões por meio de trocas sistemáticas de estruturas de vizinhança, alterando gradativamente para vizinhanças mais distantes. O VND se baseia no fato de que um ótimo local com relação a uma vizinhança não necessariamente corresponde a um ótimo com relação a outra vizinhança. Além disso, um ótimo global corresponde a um ótimo local para todas as estruturas de vizinhança e, para muitos problemas, ótimos locais com relação a uma vizinhança são relativamente próximos.

O Algoritmo 2 apresenta a heurística de busca local. Este algoritmo tem, como entrada, uma solução (s, H). Enquanto esta solução não for um ótimo local com relação às duas estruturas de vizinhança, são feitos movimentos de troca de alocação e de troca de hubs. Se estes movimentos melhorarem a solução, a melhor solução encontrada e o valor de função objetivo são atualizados.

3.5 Movimento de troca de alocação

Um movimento de troca de alocação consiste em alocar um nó não hub a um outro nó hub e desalocá-lo do hub anterior. Por exemplo, dados $N = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$,

Algorithm 2 Algoritmo de Busca Local

Require: Solução (s, H)
1: **while** (s, H) não for um ótimo local para as duas estruturas de vizinhanças **do**
2: $(\bar{s}, \bar{H}) \leftarrow$ movimento de alocação a partir de (s, H)
3: **if** $f(\bar{s}) > f(s)$ **then**
4: $(s, H) \leftarrow (\bar{s}, \bar{H})$
5: $fo \leftarrow f(\bar{s})$
6: **end if**
7: $(\bar{s}, \bar{H}) \leftarrow$ movimento de hub a partir de (s, H)
8: **if** $f(\bar{s}) > f(s)$ **then**
9: $(s, H) \leftarrow (\bar{s}, \bar{H})$
10: $fo \leftarrow f(\bar{s})$
11: **end if**
12: **end while**
13: **Retorne** (s, H)

$H = (7, 8, 9)$ e $s = (7, 7, 8, 9, 8, 9, 7, 8, 9, 9)$, uma mudança de alocação pode resultar no vetor de alocação $s_1 = (9, 7, 8, 9, 8, 9, 7, 8, 9, 9)$, em que o nó 1 deixou de ser alocado ao hub 7 para ser alocado ao hub 9.

O algoritmo que descreve esse preocedimento de busca local usando esse movimento de troca de alocação tem como entrada uma solução inicial (s, H) e seu valor de função objetivo f_o e, para cada nó $i \in N$ e para cada hub $k \in H$, é feita a troca de alocação do nó i do hub ao qual estava anteriormente alocado para um outro hub aberto. Se, com a nova alocação, o valor de função objetivo aumentar, então a solução corrente e o valor de f_o são atualizados. Caso contrario, é retomada a alocação anterior. O movimento de troca de alocação é detalhado no Algoritmo 3.

Algorithm 3 Algoritmo de troca de alocação

Require: fo e (s, H), onde $s = (s_1, \ldots, s_n)$, $H = (h_1, \ldots, h_p)$
1: **for** $i \leftarrow 1$ até n **do**
2: **for** $k \leftarrow 1$ até p **do**
3: **if** $\bar{s}_i \neq h_k$ **then**
4: $\bar{s}_i \leftarrow h_k$
5: $\bar{s}_j \leftarrow s_j, \forall\ j \neq i$
6: $\bar{f} \leftarrow f(\bar{s})$
7: **if** $\bar{f} > fo$ **then**
8: $fo \leftarrow \bar{f}$
9: $s \leftarrow \bar{s}$
10: **end if**
11: **end if**
12: **end for**
13: **end for**
14: **Retorne** (s, H)

3.6 Movimento de troca de hub

Um movimento de troca de hub consiste em fechar um hub aberto e abrir um hub que estava previamente fechado. Dada um vetor de hubs H, um vizinho da solução corrente possui vetor de hubs H_1, em que H_1 corresponde a troca de um hub de H por um nó não hub do conjunto de nós N. Por exemplo, dados $N = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ e $H = (7, 8, 9)$, ao retirar o hub 7 de H e inserir o hub 3, temos $H_1 = (8, 9, 3)$.

A solução vizinha é avaliada através da alocação gulosa e movimentos de troca de alocação da seguinte forma. Após a troca de hubs, é feita uma alocação gulosa de todos os nós de demanda, seguida da avaliação da solução resultante. Se esta solução é melhor que a anterior, ela se torna a solução corrente. Caso contrário, são feitos movimentos de troca de alocação. Se estes movimentos encontram soluções melhores, esta solução se torna a solução corrente. Caso contrário, retorna-se a configuração anterior, e tenta-se outro vizinho de mudança de hub.

O algoritmo que descreve esse procedimento de busca local usando esse movimento de troca de hub tem como entrada a solução corrente e seu valor de função objetivo f_o. Para cada nó hub h em H e para cada nó não hub i em N que não está na lista H, o nó h é trocado pelo nó i em H. Em seguida, é feita uma alocação gulosa e a solução resultante é avaliada. Se a solução encontrada tiver melhor valor de função objetivo, é feita uma atualização de f_o e da solução. Após isso, é realizada uma busca local de troca de alocação e a solução resultante é avaliada. Caso a busca local utilizando movimento de alocação resulte em uma solução melhor, é feita uma atualização de f_o e da solução. O Algoritmo 4 ilustra o procedimento de busca local usando movimento de troca de hub.

Algorithm 4 Algoritmo de Troca de Hub

Require: fo e (s, H), onde $s = (s_1, \ldots, s_n)$, $H = (h_1, \ldots, h_p)$
for $h \in H$ **do**
  for $i \in N \setminus \{h_1, \ldots, h_p\}$ **do**
    $\bar{H} \leftarrow$ Retire h de H e acrescente i em H
    $\bar{s} \leftarrow$ Alocação gulosa para $\bar{H}$
    $\bar{f} \leftarrow f(\bar{s})$
    if $\bar{f} > fo$ **then**
      $fo \leftarrow \bar{f}$
      $(s, H) \leftarrow (\bar{s}, \bar{H})$
    end if
    $\bar{s} \leftarrow$ Aplicação de movimentos de alocação na solução $(\bar{s}, \bar{H})$
    $\bar{f} \leftarrow f(\bar{s})$
    if $\bar{f} > fo$ **then**
      $fo \leftarrow \bar{f}$
      $(s, H) \leftarrow (\bar{s}, \bar{H})$
    end if
  end for
end for
Retorne (s, H)

3.7 Estratégia de busca local

Utilizamos a estratégia de busca de melhor vizinho (*Best Improvement*), que move para o melhor vizinho de melhora encontrado, após analisar todos os seus possíveis vizinhos, movendo somente para aquele que representar uma melhora no valor atual da função de avaliação.

Seja (s, H) uma solução, enquanto esta solução não for um ótimo local com relação às duas estruturas de vizinhança, são feitos movimentos de troca de alocação e de troca de hubs. Se estes movimentos melhorarem a solução, a melhor solução encontrada e o valor de função objetivo são atualizados.

3.8 Greedy Randomized Adaptive Search Procedures (GRASP)

A meta-heurística GRASP é um processo iterativo para problemas combinatórios, em que cada iteração consiste basicamente em duas fases: *Construção e Busca Local*.

O pseudocódigo representado pelo Algoritmo 5, onde $\alpha \in [0, 1]$ é um parâmetro do método e descreve a fase de construção do GRASP. Observamos que o parâmetro α controla o nível de gulosidade e aleatoriedade do procedimento *Construção*. Um valor $\alpha = 0$ faz gerar soluções puramente gulosas, enquanto $\alpha = 1$ faz produzir soluções totalmente aleatórias.

Algorithm 5 Procedimento *Construção*$(g(\cdot), \alpha, s)$

1: $s \leftarrow \emptyset$;
2: Inicialize o conjunto LC de candidatos;
3: **while** $LC \neq \emptyset$ **do**
4: $\quad g(t_{\min}) = \min\{g(t) \,|\, t \in LC\}$;
5: $\quad g(t_{\max}) = \max\{g(t) \,|\, t \in LC\}$;
6: $\quad LCR = \{t \in C \,|\, g(t) \geq g(t_{\max}) - \alpha(g(t_{\max}) - g(t_{\min}))\}$;
7: $\quad$ Selecione aleatoriamente um elemento $t \in$ LCR;
8: $\quad s \leftarrow s \cup \{t\}$;
9: $\quad$ Atualize o conjunto LC de candidatos;
10: **end while**;
11: **Retorne** s;
12: **fim** *Construção*;

O Algoritmo 6 descreve o pseudocódigo do procedimento de *Refinamento* para o nosso problema de maximização. A busca local explora as estruturas de vizinhanças através dos movimentos de troca de alocação e troca de hub, alocando nó não hub ao hub mais próximo. O pseudocódigo representado pelo Algoritmo 7 ilustra o procedimento GRASP utilizado. Este algoritmo tem, como entrada, apenas dois parâmetros, a saber: α (fator de aleatoriedade/gulosidade da fase de construção) e $GRASP_{max}$ (número de iterações em que o método é aplicado).

Neste trabalho, utilizamos $\alpha = 0,6$ e $GRASP_{max} = 30$. Na fase de construção, uma solução é iterativamente construída, elemento por elemento. A cada iteração desta fase,

os próximos elementos candidatos a serem incluídos na solução são colocados em uma lista de candidatos (LC), seguindo um critério de ordenação pré-determinado.

A seleção do próximo elemento a ser incorporado é determinada pela avaliação do todos os elementos candidatos de acordo com uma função de avaliação gulosa. Esta função gulosa geralmente representa o aumento incremental na função custo devido à incorporação deste elemento na solução em construção.

A avaliação dos elementos por esta função leva à criação de uma lista de candidatos restrita (LCR) formada pelos melhores elementos, ou seja, aqueles cuja incorporação à solução parcial atual resulta nos menores custos incrementais (este é o aspecto guloso do algoritmo). O elemento a ser incorporado na solução parcial é selecionado aleatoriamente dentre aqueles da LCR (este é o aspecto probabilístico da heurística). Uma vez incorporado o elemento selecionado à solução parcial, a lista de candidatos é atualizada e os custos incrementais são reavaliados (este é o aspecto adaptativo da heurística).

Algorithm 6 Procedimento *BuscaLocal* ($f(\cdot)$, $N(\cdot)$, s)

1: $V = \{s' \in N(s) \mid f(s') > f(s)\}$;
2: **while** $|V| > 0$ **do**
3: Selecione $s' \in V$;
4: $s \leftarrow s'$;
5: $V = \{s' \in N(s) \mid f(s') > f(s)\}$;
6: **end while**;
7: **Retorne** s;
8: **fim** *BuscaLocal*;

Algorithm 7 Procedimento *GRASP* ($f(\cdot)$, $g(\cdot)$, $N(\cdot)$, $GRASP_{\max}$, α, s)

1: $f^* \leftarrow \infty$;
2: **for** Iter $= 1$ até $GRASP_{\max}$ **do**
3: *Construção*($g(\cdot)$, α, s);
4: *BuscaLocal*($f(\cdot)$, $N(\cdot)$, s);
5: **if** $f(s) > f^*$ **then**
6: $s^* \leftarrow s$;
7: $f^* \leftarrow f(s)$;
8: **end if**
9: **end for**
10: $s \leftarrow s^*$;
11: **Retorne** s;
12: **fim** *GRASP*;

O Algoritmo 8 a seguir resume todo o procedimento aplicado ao problema USApHMCP.

Algorithm 8 Procedimento aplicado ao USApHMCP

input:Instâncias APs, α, β_{ij}

1: Leia os dados de entrada.
2: Calcular os parâmetros a_{ij}^{km} de acordo com o (Algorithm 1).
3: Para cada nó $i \in N$, calcule o fluxo total com destino (D_i) e com origem (O_i) no nó i: $O_i = \sum_{j \in N} W_{ij}$ e $D_i = \sum_{j \in N} W_{ji}$.
4: Ordene os nós $i \in N$ em ordem decrescente de $(O_i + D_i)$.
5: Selecione os p primeiros nós com maior fluxo de entrada e saída como hubs.
6: Gere uma solução inicial usando alocação por menor distância e calcule o valor de função objetivo.
7: Busca local (Algorithm 2, Algorithm 3 - Algorithm 4).
8: Meta-heurística GRASP (Algorithm 5, Algorithm 6 - Algorithm 7).

4 Resultados dos Testes Computacionais

Todos os experimentos computacionais deste trabalho foram realizados em um notebook com um sistema operacional Windows 11 com processador 11^a Geração do Intel(R) Core(TM) $i7-1165G7$ 2.80 GHz e 16 GB de memória RAM.

Para a obtenção dos resultados computacionais foi utilizado um conjunto de dados conhecido na literatura como *Australian Post* (AP), com $n = 10, 20, 25, 40$ nós, $p = 3$, $\gamma = \delta = 1$ e $\alpha = 0,75$.

Como os resultados não foram satisfatórios, foram utilizadas para teste apenas 4 instâncias do conjunto AP. Neste conjunto, APn_pL indica uma instância com n nós e p hubs. Para os testes, também foi utilizada a versão acadêmica do software IBM ILOG CPLEX, versão 20.1.0.0. Foram coletados os tempos de execução e os valores de f_o tanto do CPLEX, como do algoritmo implementado. O tempo de execução não foi utilizado como critério de parada, ou seja, para o caso do CPLEX, foi encontrado o valor final de f_o.

Os resultados para $b = \beta_{ij} = 2609$ e $b = \beta_{ij} = 25095$ são mostrados nas Tabelas 1 e 2, respectivamente. Nestas tabelas, $fo\ CPLEX$ é o valor de função objetivo encontrado pelo CPLEX; t_{CPLEX} é o tempo de execução do CPLEX; $fo\ Heur$ é o valor de função objetivo da melhor solução encontrado pela heurística; e t_{Heur} é o tempo de execução da heurística. O gap para a função objetivo, dado por:

$$gap = 100\frac{fo\ CPLEX - fo\ Heur}{fo\ Heur}$$

é mostrado em valores percentuais em relação ao valor encontrado pelo algoritmo heurístico. O tempo de execução é mostrado em segundos.

Durante a execução dos algoritmos nenhum outro processo foi executado paralelamente na máquina.

Observando as Tabelas 1 e 2, pode-se notar que a heurística não encontrou as soluções ótimas para nenhuma das instâncias utilizadas. Os gaps foram relativamente altos para todas as instâncias testadas. Por isso, paramos de testar outras instâncias, visto que ajustes na implementação devem ser realizados.

Tabela 1: Resultados para $\beta_{ij} = 2609$.

Instâncias	*fo CPLEX*	t_{CPLEX}	*fo Heur*	t_{Heur}	*GAP*
$AP10_3L$	477,66	0,03	362,146	0,12	31,89
$AP20_3L$	247,689	0,06	188,287	0,19	31,54
$AP25_3L$	352,841	0,09	247,277	0,20	42,69
$AP40_3L$	302,05	0,19	208,31	0,21	45

Tabela 2: Resultados para $\beta_{ij} = 25095$.

Instâncias	*fo CPLEX*	t_{CPLEX}	*fo Heur*	t_{Heur}	*GAP*
$AP10_3L$	3031,81	0,23	2638.61	0.069	14,9
$AP20_3L$	2915,95	3,3	2366.22	0,826	23,23
$AP25_3L$	2829,16	8,45	2555.25	1,479	10,71

5 Análise dos Resultados e Considerações Finais

Conforme visto na seção anterior, os resultados não foram satisfatórios. Portanto, ajustes no algoritmo serão realizados afim de encontrar boas soluções em tempo computacional melhor que o CPLEX.

Para trabalhos posteriores, pretende-se rever a implementação visando melhorar o valor de função objetivo, melhorando assim o GAP; testar outras instâncias da literatura e calibrar os parâmetros com o IRACE.

6 Referências Bibliográficas

Almeida, Wesley G., Senne, Edson L. F., Yanasse, Horacio H. *Abordagens meta-heurísticas para o problema de localização de concentradores com restrições de capacidade.* Anais [...] São José dos Campos: INPE, 2010. On-line. IBI:8JMKD3MGP8W/38BJHQ5.Disponível em:http://urlib.net/ibi/8JMKD3MGP8W/38BJHQ5.

Butinholi, Matheus Martins, Alexandre Barcellos de Oliveira, Paganini Martino, Diego. (2020). *Basic VNS for the Uncapacitated Single Allocation p-Hub Maximal Covering Problem.* 10.1007/978-3-030-44932-29.

Campbell, J. F. (1994). *Integer programming formulations of discrete hub location problems.* European Journal of Operational Research, 72(2):387–405.

O'Kelly, M. E., Miller, H. J., 1994. *The hub network design problem: A review and synthesis.* Journal of Transport Geography 2 (1), 31–40.

Peker, M. e Kara, B. Y. (2015). *The p-hub maximal covering problem and extensions for gradual decay functions.* Omega, 54:158–172.

Silva, Fernando; Sá, Elisângela M.; Souza, Sérgio R. *Uma meta-heurística Iterated Local Search para o problema p-hub de máxima cobertura não capacitado com alocação única.* In: Anais do Simpósio Brasileiro de Pesquisa Operacional, 2022, Juiz de Fora.

Silva, M. R. e Cunha, C. B. (2017). *A tabu search heuristic for the uncapacitated single allocation p-hub maximal covering problem.* European Journal of Operational Research, 262:954–965.

Yang, K., Liu, Y.-K., e Yang, G.-Q. (2013). *Solving fuzzy p-hub center problem by genetic algorithm incorporating local search.* Applied Soft Computing, 13(5):2624–2632.