Metaheurística *Iterated Local Search* Aplicada ao Problema de Localização com Cobertura Parcial

Leonardo Correa Cardoso[1], **Fábio Pires Mourão**[1,2], **Elisangela Martins de Sá**[1], **Sérgio Ricardo de Souza**[1]

[1]Programa de Pós-Graduação em Modelagem Matemática e Computacional
Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Av. Amazonas 7675, CEP 30.510-000, Belo Horizonte (MG), Brasil

[2]Instituto Federal de Educação Ciência e Tecnologia de Minas Gerais (IFMG)
R. Itaguaçu, 595, Bairro São Caetano, Betim/MG

lccardoso13@gmail.com,fabio.mourao@ifmg.edu.br

elisangelamartins@cefetmg.br, sergio@cefetmg.br

Abstract. *The partial set covering location problem consists of locating a set of facilities to minimize the total location cost, ensuring that a predetermined amount of customer demand is covered by these facilities. This paper proposes an algorithm based on the metaheuristic Iterated Local Search for solving this problem. Furthermore, a set of computational experiments were performed and the results of these experiments show that good solutions can be found for moderately large instances.*

Resumo. *O problema de localização de cobertura de conjunto parcial consiste em localizar um conjunto de instalações de forma a minimizar o custo total de localização e garantir que uma quantidade predeterminada de demanda de clientes seja coberta por estas instalações. Este artigo apresenta um algoritmo baseado na metaheurística Iterated Local Search para a resolução deste problema. Além disso, um conjunto de experimentos computacionais foram realizados e resultados demonstram que boas soluções podem ser encontradas para instâncias moderadamente grandes.*

1. Introdução

Problemas de localização de cobertura fazem parte de um grupo de problemas de localização de grande importância. Estes problemas consistem em determinar a localização de instalações de forma a cobrir demandas de clientes. Neste tipo de problemas, em geral, um cliente é considerado coberto por uma instalação se a distância entre os dois for menor ou igual a uma distância máxima previamente definida, denominada raio de cobertura.

Na literatura, encontram-se diversas aplicações para problemas de cobertura. Em [Church and ReVelle 1974] é proposto o problema de máxima cobertura (PMC), conhecido, em inglês, por *Maximal Covering Location Problem*. Este problema tem como objetivo maximizar a cobertura de uma certa população, considerando um número fixo

de instalações candidatas. Os autores apresentaram ainda uma curva de variação, realizando uma análise entre o número de instalações elegíveis para fornecer cobertura e a quantidades da população coberta à medida que adiciona-se novas instalações à solução corrente. [Galvão and ReVelle 1996] apresentam uma heurística lagrangeana para o mesmo problema e os resultados de testes computacionais para redes com até 150 nós. [Coco et al. 2018] apresentam uma abordagem robusta para lidar com a incerteza nos dados de entrada do PMC.

[Daskin and Owen 1999] introduzem duas novas variações para problemas de cobertura. Eles apresentaram o problema p-centro de cobertura parcial, que minimiza a distância de cobertura de tal forma que uma determinada fração da população é coberta. Foi apresentado também o problema de localização de cobertura de conjunto parcial, denominado, em inglês, por *partial set covering location problem* (PSCLP), que busca minimizar o custo de abertura das instalações necessárias para cobrir uma certa fração da população. Nesse trabalho, os autores tiveram, como objetivo, preencher a lacuna que até então existia para o tema na literatura, demonstrando que os modelos propostos podem servir de base para um processo de planejamento logístico no qual pode ser impossível ou antieconômico atender todos os clientes igualmente bem. Além disso, pode-se identificar locais com clientes que são particularmente difíceis de atender.

Neste trabalho, é apresentado um estudo com foco no PSCLP. Este problema também foi abordado por [Bilal et al. 2014], que propõem uma heurística *Iterated Tabu Search* para uma variação do problema que tem, como objetivo, maximizar o lucro, porém, não sendo necessário cobrir toda a demanda. O algoritmo proposto usa dois níveis de perturbação e uma heurística de busca tabu. [Mišković 2017] propõe uma variação robusta para o problema de cobertura máxima dinâmica, denominado em inglês por *dynamic maximal covering location problem*. O problema é resolvido através da metaheurística *Variable Neighborhood Search - Linear Program* (VNS-LP), baseada na hibridização de uma busca em vizinhança variável e uma técnica de programação linear. [Cordeau et al. 2019] apresenta uma abordagem para o PSCLP utilizando o método de decomposição de Benders com foco em reduzir tempos computacionais para resolução de instâncias grandes do problema.

Como o PSCLP é um problema NP-difícil, para resolver instâncias de maior dimensão do problema é necessário o uso de procedimentos heurísticos. Portanto, este trabalho propõe um algoritmo baseado na metaheurística *Iterated Local Search* (ILS). A partir de experimentos computacionais utilizando o conjunto de instâncias utilizadas no trabalho de [Cordeau et al. 2019], é feita um comparação do tempo de execução e da qualidade das soluções obtidos pelo algoritmo proposto e pelo *solver* CPLEX.

Este trabalho esta organizado da seguinte maneira. A Seção 2 apresenta uma caracterização do problema. A Seção 3 descreve a metodologia utilizada para a solução do modelo matemático. A Seção 4 apresenta as ferramentas computacionais utilizadas, bem como os resultados computacionais obtidos após a implementação e testes com um conjunto de instâncias da literatura. Por fim, a Seção 5 apresenta as conclusões e propostas para trabalhos futuros.

2. Caracterização do Problema

O problema de localização com cobertura parcial tem, como objetivo, minimizar o custo de abertura de novas instalações, ao mesmo tempo em que deve garantir o atendimento de uma demanda mínima de clientes. Seja I o conjunto formado por n instalações candidatas e J um conjunto com m clientes. Cada cliente $j \in J$ possui uma demanda d_j e cada instalação $i \in I$ possui um custo de abertura f_i. Há, também, uma demanda mínima a ser atendida, dada por D. Uma representação das possíveis coberturas de clientes por instalações pode ser feita por uma matriz de cobertura A_{nxm}. Esta matriz possui elementos $a_{ij} \in \{0, 1\}$, em que $a_{ij} = 1$ caso o cliente j esteja dentro do raio de cobertura da instalação i, e $a_{ij} = 0$, caso contrário. A Figura 1 apresenta um exemplo de matriz de cobertura para este problema, em que arestas conectando clientes a instalações representam possíveis coberturas. De acordo com essa figura, a instalação 1 cobre o cliente 1, porém a mesma instalação não cobre o cliente 4. Assim, tem-se, na matriz de cobertura, a posição $a_{11} = 1$ e a posição $a_{14} = 0$.

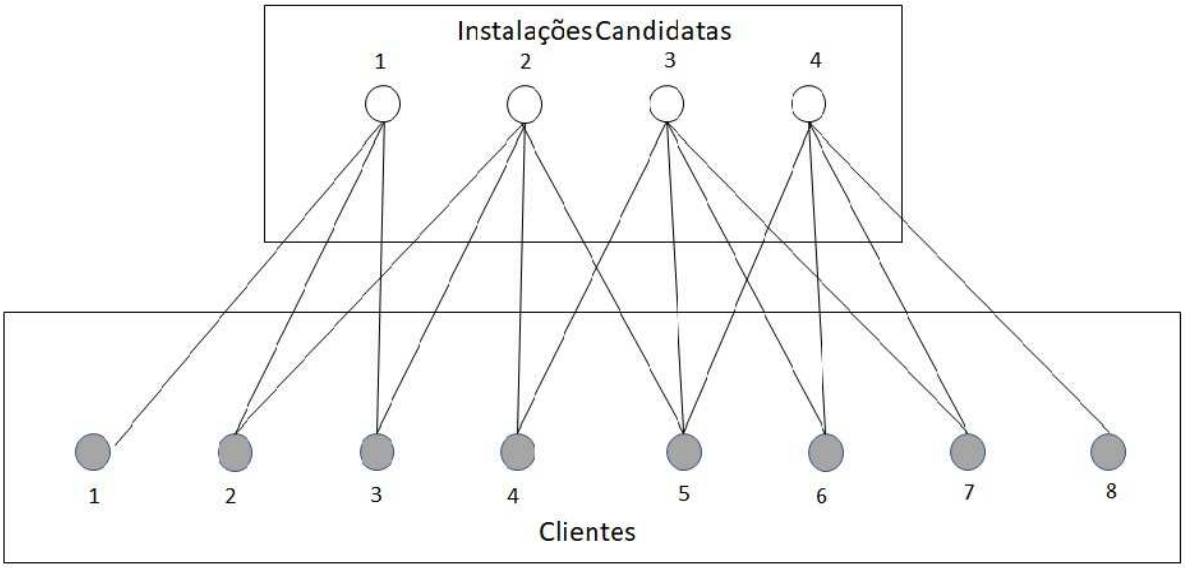

Figura 1. Representação gráfica de relações de possíveis coberturas para uma instância com 4 instalações em potencial e 8 clientes.

Seja y_i uma variável binária tal que:

$$y_i = \begin{cases} 1 & \text{se a instalação } i \in I \text{ estiver aberta} \\ 0 & \text{caso contrário.} \end{cases}$$

e z_j uma variável binária tal que:

$$z_j = \begin{cases} 1 & \text{se o cliente } j \in J \text{ for coberto por pelo menos uma instalação} \\ 0 & \text{caso contrário.} \end{cases}$$

Pode-se formular o PSCLP da seguinte forma:

$$\min \sum_{i=1}^{n} f_i y_i \tag{1}$$

$$s.a \sum_{i=1}^{n} a_{ij} y_i \geq z_j \qquad \forall j \in J \tag{2}$$

$$\sum_{j=1}^{m} d_j z_j \geq D \tag{3}$$

$$y_i \in \{0,1\} \qquad \forall i \in I \tag{4}$$

$$z_j \in \{0,1\} \qquad \forall j \in J \tag{5}$$

A função objetivo (1) minimiza o custo de abertura das instalações. As restrições (2) garantem que, caso um cliente j esteja coberto, haverá pelo menos uma instalação aberta dentro do seu raio de cobertura o cobrindo. A restrição (3) garante que o mínimo de demanda D seja atendido. Finalmente, as restrições (4) e (5) impõem restrições binárias às variáveis de decisão y e z, respectivamente.

3. Metodologia

Esta seção apresenta a metodologia utilizada para resolução de instâncias do PSCLP, incluindo a representação da solução, as estruturas de vizinhanças exploradas, bem como as heurísticas utilizadas.

3.1. Representação da solução

A representação de uma solução para o problema é feita por dois vetores, com papel semelhante ao das variáveis de decisão do modelo matemático, denotados também por y e z. O vetor y é um vetor binário de dimensão n (número de instalações candidatas) em que o valor 1 em uma dada posição i representa que a instalação i está aberta e o valor 0 representa que a instalação i está fechada. Um exemplo para o vetor y para uma solução de um problema com $n = 6$ instalações candidatas e 3 instalação abertas é dado por:

$$y = [1, 1, 0, 0, 1, 0], \tag{6}$$

As posições 1, 2 e 5 do vetor representam as instalações abertas e as posições 3, 4 e 6 representam as instalações fechadas.

O vetor z representa os clientes cobertos, em que o valor 1 em uma dada posição j representa que o cliente j é coberto por uma instalação ativa, e o valor 0, caso contrário. Uma ilustração do vetor z para uma solução em que $m = 10$ seria o vetor:

$$z = [1, 1, 0, 0, 1, 0, 1, 1, 0, 0], \tag{7}$$

em que apenas os clientes 1, 2, 5, 7 e 8 são cobertos.

3.2. Função de Avaliação

A função de avaliação é representada pela própria função objetivo do problema. Seja $s = (y, z)$ uma solução com vetores de representação da solução y e z, como descritos anteriormente. Então, o valor da função de avaliação para a solução s será denotado por $f(s)$, que é dada por:

$$f(s) = \sum_{i=1}^{n} f_i y_i. \tag{8}$$

3.3. Construção da Solução Inicial

A estratégia utilizada para a construção de uma solução inicial foi uma construção por inserção mais barata. Primeiramente, todas as instalações candidatas foram ordenadas, avaliando o custo de abertura das mesmas e as ordenando do menor custo para o maior. Cada instalação está associada ao conjunto de clientes pela relação de possíveis coberturas estabelecida por um raio de cobertura.

O Algoritmo 1 ilustra como a solução inicial é construída. Inicialmente, é aberta, como primeira instalação, a instalação com o menor custo de abertura. Feita esta abertura, verifica-se, de acordo com o raio de cobertura, todos os clientes que podem ser atendidos por esta instalação, considerando-se estes clientes como cobertos. Como cada cliente está associado a uma demanda a ser atendida, as demandas destes clientes definem a demanda total atendida pela solução corrente. Caso a demanda total atendida pela solução corrente seja maior ou igual a D, o procedimento é encerrado, pois se tem uma solução viável. Caso contrário, é realizada a ativação da próxima instalação de menor custo de abertura e repete-se novamente todo o procedimento, até que o critério de aceitação para a demanda do problema seja satisfeita. Ao finalizar este procedimento, a solução construída é retornada. Neste algoritmo, a notação $\bar{0}$ representa uma vetor nulo com as dimensões adequadas aos vetores y e z.

Algorithm 1 ConstrucaoGulosa () - Inserção mais barata

1: $y \leftarrow \bar{0}$
2: $z \leftarrow \bar{0}$
3: $\bar{d} \leftarrow 0$
4: $C \leftarrow$ Conjunto de instalações ordenado pelo menor custo de abertura
5: **while** $\bar{d} < D$ **do**
6: $\quad i \leftarrow$ Instalação com menor custo de abertura em C
7: $\quad C \leftarrow C \setminus \{i\}$
8: $\quad y_i \leftarrow 1$
9: $\quad z_j \leftarrow 1$, para todo j tal que $a_{ij} = 1$
10: $\quad \bar{d} \leftarrow$ demanda total coberta pelas instalações abertas
11: **end while**
12: Retorne $s_0 = (y, z)$;
13: **fim** ConstrucaoGulosa;

3.4. Heurística de Refinamento

Para explorar o espaço de soluções, foi utilizado o método de Descida em Vizinhança Variável (*Variable Neighborhood Descent* – VND), proposto por [Mladenović and Hansen 1997]. O método VND explora o espaço de soluções por meio de trocas sistemáticas de estruturas de vizinhança, explorando, gradativamente, vizinhanças cada vez mais "distantes". Para o problema proposto, foram utilizadas duas estruturas de vizinhança, baseadas em dois movimentos.

O primeiro movimento é o movimento de trocas de bits, em que ocorre a abertura ou o fechamento de instalações. Este movimento tem a vantagem de fornecer a garantia de exploração do todo o espaço de soluções para o problema. O segundo movimento é o de troca de instalações, que fecha uma instalação aberta e abre uma instalação fechada. Porém, com esta estrutura de vizinhança não se tem a garantia da exploração de todo o espaço solução, já que a quantidade de instalações abertas é fixo. No entanto, em algumas situações não é possível encontrar uma solução viável melhor apenas fechando ou abrindo uma única instalação.

O pseudocódigo no Algoritmo2 descreve a implementação realizada do método VND. O algoritmo parte de uma solução inicial s_0 e considera r estruturas de vizinhanças, que, neste caso, será igual a 2. As estruturas de vizinhanças são denotadas por N^k para $k = 1, 2$. As estruturas mencionados foram ordenadas da seguinte forma: (i) troca de bits; seguida por (ii) troca de instalações. Assim, em cada iteração, é feita uma busca local na solução corrente através da primeira estrutura de vizinhança, ou seja, vizinhança baseada na troca de bits. Caso seja encontrada uma solução melhor, a variável s é atualizada e a busca continua nesta estrutura de vizinhança. Caso contrário, finaliza-se a busca nesta estrutura de vizinhança e se inicia uma nova busca na estrutura de vizinhança referente à troca de instalações. Nesta etapa, caso ocorra melhoria na solução corrente s, esta solução é atualizada e a busca retorna para a primeira estrutura de vizinhança. Caso contrário, o algoritmo se encerra, retornando uma solução que é ótimo local com relação às duas estruturas de vizinhança. A exploração das duas estruturas de vizinhanças é baseada da estratégia de melhor vizinho, em que a vizinhança completa da solução corrente é explorada.

3.5. Algoritmo ILS

A Metaheurística *Iterated Local Search* (ILS), descrita no Algoritmo 3, é adotada para a solução do problema. O procedimento ILS [Lourenço et al. 2003] é um método que realiza busca local em novas soluções geradas por meio de perturbações na solução ótima local corrente.

Inicialmente, é gerada uma solução inicial, que é atribuída à variável s_0, conforme apresentado na linha 1. Em seguida, faz-se uma busca local na solução s_0, atribuindo-se à solução melhorada a variável s, sendo esta então a solução corrente. Após, são inicializadas as variáveis $iter$, utilizada para realizar a contagem do número de iterações, e h, que contabiliza o número de iterações sem melhora. Inicia-se, então, o laço de repetição condicional, no qual é realizada uma perturbação na solução corrente, conforme será detalhado logo mais. Posteriormente, usando o método VND, apresentado no Algoritmo 2, é realizada a busca local na solução perturbada s' encontrada, dando origem a uma solução s''. Em seguida, é verificado o critério de aceitação. Se a solução s'' for melhor que a

Algorithm 2 VND - *Variable Neighborhood Descent*

Seja s_0 uma solução inicial e r o número de estruturas de vizinhança
$s \leftarrow s_0$
$k \leftarrow 1$
while $k \leq r$ **do**
    Encontre o melhor vizinho $s' \in N^{(k)}(S)$;
    if $(f(s') \leq f(s))$ **then**
        $s \leftarrow s'$
        $k \leftarrow 1$;
    else
        $k \leftarrow k + 1$;
    end if
end while
Retorne s;
fim VND;

solução corrente s, a solução s é atualizada para s'' e o número de iterações sem melhora é zerado. Caso contrário, a solução s continua sendo igual à solução corrente e o número de iterações sem melhora é incrementado. Este procedimento é repetido enquanto o número de iterações sem melhoras for menor ou igual a $ILS_{\max}$, que é um parâmetro do algoritmo.

O procedimento de perturbação é apresentado no Algoritmo 4, que consiste na aplicação de uma quantidade de movimentos aleatórios de troca de bits. O parâmetro $nivel$ define a quantidade de movimentos realizada.

Algorithm 3 $ILS(ILS_{\max}, nivel)$

$s_0 \leftarrow ConstrucaoGulosa()$
$s \leftarrow BuscaLocal(s_0)$
$iter \leftarrow 0$;
$h \leftarrow 0$
while $h \leq ILS_{\max}$ **do**
    $iter \leftarrow iter + 1$
    $s' \leftarrow perturbacao(s, nivel)$
    s" $\leftarrow BuscaLocal(s')$
    $(s, h) \leftarrow CriteriosAceitacao(s, s$")
end while
Retorne s;
fim ILS;

4. Resultados computacionais

O procedimento ILS foi implementado em C++, considerando o parâmetro $ILS_{\max} = 100$ e o parâmetro $nivel = 2$ para as instâncias com 10.000 clientes e igual a 5 para o conjunto de instâncias com 50.000 clientes. Estes valores de parâmetros foram definidos empiricamente, após a realização de testes e se verificar a qualidade dos resultados obtidos, bem como os tempos de execução para cada solução. Para os experimentos

Algorithm 4 Procedimento de Perturbação

1: **perturbação** (s, nível)
2: $s' \leftarrow s$;
3: $nModificacoes \leftarrow nivel + 1$;
4: $cont \leftarrow 1$;
5: **while** $(cont < nModificacoes)$ **do**
6: $\quad s' \leftarrow$ Realiza um movimento aleatório de troca de bits em s'
7: $\quad cont \leftarrow cont + 1$
8: **end while**
9: retorne s'

computacionais, foi utilizado um computador com sistema operacional Windows 10 Pro, processador Intel Core I5 9ª geração 8250U CPU@1.60GHz e 8GB de memória RAM. Como forma de comparar os resultados do algoritmo proposto, o modelo matemático dado pelas expressões (1)-(5) foi implementado usando a linguagem Concert Technology em C++ e resolvido através do *solver* CPLEX.

Foram realizados testes computacionais com instâncias com 10.000 e 50.000 clientes, do mesmo conjunto de instâncias utilizado no trabalho de [Cordeau et al. 2019], variando o raio de cobertura e o percentual de demanda a ser atendida, mantendo sempre o número de 100 instalações em potencial. Os resultados são apresentados na Tabela 1.

Nesta Tabela, a primeira coluna mostra o número de clientes; a segunda coluna o percentual da demanda a ser atendido; a terceira coluna mostra o raio de cobertura. A quarta coluna mostra o valor encontrado pelo solver CPLEX quando soluciona a instância indicada, enquanto a quinta coluna mostra o tempo de execução computacional associado, em segundos. A sexta coluna apresenta o valor encontrado pela metaheurística baseada em ILS ao solucionar a instância indicada, e sétima coluna mostra o tempo de execução computacional associado, em segundos. A oitava e última coluna mostra a diferença percentual entre os resultados obtidos pelo CPLEX e pelo algoritmo ILS proposto. O cálculo desta diferença (nomeada como *gap%*) é dado por:

$$gap\% = 100 \left(\frac{fo(ILS) - fo(CPLEX)}{fo(ILS)} \right). \tag{9}$$

Foram solucionadas 26 instâncias. O valor de gap% foi nulo em 16 destas instâncias, ou seja, em $62,5\%$ das instâncias o procedimento proposto encontrou a solução ótima. Dentre as 13 instâncias de 10.000 clientes avaliadas, em 9 (ou seja, $69,2\%$) destas o valor ótimo foi encontrado. Por outro lado, dentre as 13 instâncias envolvendo 50.000 clientes, o valor ótimo foi determinado para 7 instâncias ($53,8\%$). Ou seja, os resultados são de boa qualidade. Quanto ao tempo computacional, para algumas instâncias os tempos computacionais gastos pelo algoritmo proposto são menores que os tempos gastos pelo solver CPLEX. Uma possível explicação para o custo computacional mais elevado nestas instâncias é o uso da técnica de busca local do melhor vizinho, em que foi feita a exploração de todas as vizinhanças disponíveis, sendo, portanto, uma estrutura dispendiosa, mas que garante a determinação de melhor resultado na vizinhança avaliada.

Tabela 1. Comparação entre os resultados obtidos pela metaheurística ILS (fo(ILS)) e os resultados obtidos pelo CPLEX (fo(CPLEX)) para a solução de instâncias do Problema de Cobertura Parcial.

N Clientes	Demanda(%)	Raio	fo(CPLEX)	Tempo (seg)	fo(ILS)	Tempo (seg)	gap%
10.000	50	5,50	74,00	4,95	74,00	2,80	0,00
		5,75	63,00	5,66	63,00	2,78	0,00
		6,00	60,00	5,02	60,00	2,79	0,00
		6,25	48,00	2,34	48,00	3,13	0,00
10.000	60	4,00	207,00	15,13	209,00	9,10	0.97
		4,25	176,00	4,78	176,00	8,33	0,00
		4,50	154,00	8,72	154,00	9,70	0,00
		4,75	133,00	6,44	133,00	9,66	0,00
		5,00	120,00	7,31	120,00	5,28	0,00
10.000	70	3,25	517,00	3,42	537,00	24,96	3,87
		3,50	408,00	3,95	418,00	17,57	2,45
		3,75	324,00	7,84	333,00	12,48	2,78
		4,25	238,00	6,66	238,00	15,72	0,00
50.000	50	5,50	82,00	53,22	88,00	39,68	7,32
		5,75	76,00	47,36	76,00	30,63	0,00
		6,00	68,00	19,38	75,00	23,33	10,29
		6,25	66,00	213,14	66,00	21,47	0,00
50.000	60	4,00	248,00	183,23	248,00	123,44	0.00
		4,25	217,00	187,52	217,00	110,75	0,00
		4,50	185,00	33,92	189,00	143,23	2,16
		4,75	161,00	28,18	168,00	67,29	4,35
		5,00	137,00	30,08	137,00	63,03	0,00
50.000	70	3,25	611,00	321,15	636,00	412,57	4,09
		3,50	478,00	10,75	491,00	226,21	2,72
		3,75	325,00	138,17	325,00	145,30	0,00
		4,25	284,00	256,63	284,00	101,06	0,00

5. Conclusão

O projeto apresentado está em fase inicial de desenvolvimento e ainda assim os métodos propostos obtiveram soluções ótimas para o conjunto de instâncias testadas. Porém os tempos de execução obtidos foram maiores que os dos tempos do CPLEX para algumas instâncias. Conforme discutido na seção anterior isso provavelmente se deve ao método de busca local utilizado na solução do problema. Como forma de melhorar os tempos de execução será implementado o método de busca local *First Improvement* buscando evitar a busca exaustiva pelo melhor vizinho e encerrando a exploração da vizinhança logo que um melhor vizinho é encontrado, garantindo assim que a solução encontrada é um ótimo local com relação a vizinhança pesquisada. Posteriormente sendo possível utilizar o ILS para fugir desses ótimos locais aumentando assim a exploração do espaço de solução e a busca pelo ótimo global para o problema.

Agradecimentos

Os autores agradecem ao Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG) e ao Instituto Federal de Educação Ciência e Tecnologia de Minas Gerais (IFMG) - Campus Betim pelo apoio ao desenvolvimento do presente estudo.

Referências

Bilal, N., Galinier, P., and Guibault, F. (2014). An iterated-tabu-search heuristic for a variant of the partial set covering problem. Journal of Heuristics, 20(2):143–164.

Church, R. and ReVelle, C. (1974). The maximal covering location problem. In Papers of the regional science association, volume 32, pages 101–118. Springer-Verlag.

Coco, A. A., Santos, A. C., and Noronha, T. F. (2018). Formulation and algorithms for the robust maximal covering location problem. Electronic Notes in Discrete Mathematics, 64:145–154.

Cordeau, J.-F., Furini, F., and Ljubić, I. (2019). Benders decomposition for very large scale partial set covering and maximal covering location problems. European Journal of Operational Research, 275(3):882–896.

Daskin, M. S. and Owen, S. H. (1999). Two new location covering problems: The partial p-center problem and the partial set covering problem. Geographical Analysis, 31(3):217–235.

Galvão, R. D. and ReVelle, C. (1996). A lagrangean heuristic for the maximal covering location problem. European Journal of Operational Research, 88(1):114–123.

Lourenço, H. R., Martin, O. C., and Stützle, T. (2003). Iterated Local Search, pages 320–353. Springer US, Boston, MA.

Mišković, S. (2017). A VNS-LP algorithm for the robust dynamic maximal covering location problem. OR Spectrum, 39(4):1011–1033.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. Computers & Operations Research, 24(11):1097–1100.