HEURÍSTICA HÍBRIDA GRASP/ILS PARA O PROBLEMA DE LOCALIZAÇÃO DE p INSTALAÇÕES INDESEJADAS

Wesley de Matos Lancuna
Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Av. Amazonas 7675, 30510-000 – Nova Gameleira – Belo Horizonte – MG – Brasil
wesleylancuna@yahoo.com.br

Elisangela Martins de Sá
Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Av. Amazonas 7675, 30510-000 – Nova Gameleira – Belo Horizonte – MG – Brasil
elisangelamartins@cefetmg.br

Sérgio Ricardo de Souza
Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Av. Amazonas 7675, 30510-000 – Nova Gameleira – Belo Horizonte – MG – Brasil
sergio@dppg.cefetmg.br

RESUMO

Este artigo tem seu foco no problema de localização de instalações indesejadas. O problema consiste em localizar instalações, de modo que as mesmas estejam o mais afastado possível dos clientes. As instalações serão selecionadas de forma a maximizar a soma das distâncias dos clientes à instalação mais próxima. Possíveis aplicações desse problema são instalações de aterros sanitários, usinas nucleares, barragens de rejeitos de minério e penitenciárias. O problema é considerado NP-difícil. Para buscar melhores soluções para o problema, propõe-se um algoritmo híbrido que combina as meta-heurísticas GRASP e ILS. Os resultados mostram que a técnica proposta apresenta bons resultados comparada a algoritmos da literatura em relação à qualidade das soluções e tempo computacional.

PALAVRAS CHAVE. Problema de Localização de Instalações Indesejadas, GRASP, ILS.

Tópicos: Metaheurísticas.

ABSTRACT

This article focuses on the obnoxius facility location problem. The problem consists in locating facilities so that they are as far away as possible from the customers. The facilities are selected so to maximize the sum of the distances of the clients to the nearest facility. Possible applications of this problem are landfill facilities, nuclear power plants, ore tailings dams and prisons. The problem is considered NP-difficult. To solve the problem, we propose a heuristic algorithm that combines GRASP and ILS techniques. The results show that the proposed heuristic is equivalent to the best algorithms in the literature regarding solution quality and computational time.

KEYWORDS. Obnoxious p-Median Problem, GRASP, ILS.

Paper Topics: Metaheuristics.

1. Introdução

Neste trabalho é estudado o Problema de Localização de p Instalações Indesejadas – p-PLII (*Obnoxious p-Median Problem*). O p-PLII é uma adaptação do problema clássico de p-medianas. O problema clássico de localização de p-medianas consiste em localizar p instalações, com o intuito de minimizar a soma ponderada das distâncias de cada cliente à instalação mais próxima. No p-PLII, por outro lado, deseja-se localizar p instalações o mais afastado possível dos clientes. Possíveis aplicações para este problema são instalações de aterros sanitários, usinas nucleares, barragens de rejeitos de minério, penitenciárias, dentre outras.

Para Hosseini e Esfahani [2009], em geral, as instalações são divididas em dois grupos. As instalações do primeiro grupo são de interesse e desejáveis para os habitantes próximos, que gostariam de tê-las próximas, como hospitais, postos de bombeiros, lojas de compras e centros educacionais. O segundo grupo de instalações se refere à instalações que são indesejáveis para a população circunvizinha, que as evita, buscando ficar o mais longe possível, como depósitos de lixo, fábricas de produtos químicos, reatores nucleares, instalações militares e presídios. Devido a questões de saneamento, segurança ou bem-estar, instalações como estas não são desejáveis e tenta-se, do ponto de vista de planejamento de suas localizações, afastá-las dos centros de demanda. Observe que, apesar destas instalações indesejáveis serem, em geral, necessárias para a comunidade, como o caso de locais de despejo de lixo, a localização de tais instalações pode ser indesejada para a população ao seu redor.

Formalmente, o p-PLII consiste em localizar p instalações, dentre um conjunto de instalações candidatas, de forma a maximizar a soma da distância entre os nós de demanda e as instalações mais próximas a cada um deles. Neste caso, assume-se que a instalação mais próxima do cliente é a instalação que provoca maior efeito negativo.

Neste trabalho é proposto um algoritmo híbrido GRASP-ILS para buscar melhores soluções para instâncias do problema p-PLII. Este algoritmo é composto por duas fases: (i) fase construtiva e (ii) busca local. A fase construtiva da metaheurística GRASP Resende e Ribeiro [1998] gera várias soluções iniciais. Em seguida, após várias construções, é feita uma busca local na melhor solução construída pelo GRASP usando ILS [Lourenço et al., 2003].

Este trabalho está organizado da seguinte forma. Na Seção 2 é feita uma breve revisão da literatura, citando os principais trabalhos relacionados ao p-PLII. Na Seção 3 descreve-se, formalmente, o problema. A heurística proposta é apresentada na Seção 4. São apresentados resultados dos experimentos computacionais obtidos pela heurística e comparados com algoritmos da literatura na Seção 5. A Seção 6, finalmente, apresenta as conclusões advindas do trabalho e discorre a respeito de possíveis trabalhos futuros.

2. Revisão da Literatura

Conforme citado em Batta e Chiu [1988], o problema p-PLI foi introduzido por Goldman e Dearing [1975]. Batta e Chiu [1988] apresentam um modelo de transporte de material perigoso em uma rede na qual a soma das distâncias de um contêiner dentro de um limite de demanda de centros é minimizada. Gopalan et al. [1990] apresentaram uma heurística para o problema de localização indesejada. Já Melachrinoudis et al. [1995] abordam um problema específico de localização de aterros sanitários, desenvolvendo um modelo de otimização inteira mista multiobjetivo. Uma abordagem de localização contínua de instalações indesejáveis dentro de uma determinada região geográfica, considerando os aspectos ambientais, é proposta por Fernández et al. [2000]. Cappanera et al. [2003] abordou o problema de localizar simultaneamente instalações indesejadas e encaminhar materiais indesejados entre um conjunto de áreas construídas. Rakas et al. [2004] abordam uma modelagem multiobjetivo com critérios técnicos de eliminação para determinar a localização de instalações indesejadas. A maioria das pesquisas sobre a localização de instalações indesejadas concentrou-se na dispersão das instalações, sem considerar as interações entre instalações e clientes (ou outras instalações existentes e centros populacionais). No entanto, os modelos podem ser irrealistas, ao ignorarem os efeitos de instalações indesejadas em centros populacionais ou próximos a clientes [Chiang e Lin, 2017].

A variante do problema p-PLII usada nesta trabalho, foi proposta por Labbé et al. [2001] e pertence a classe dos problemas NP-Difícil conforme demonstrado por Tamir [1991]. Para a solução desta variante, Labbé et al. [2001] propõem um algoritmo *branch-and-cut*; porém, só obtiveram resultados satisfatórios para instâncias de pequena dimensão, com quantidade de clientes n até 75. Belotti et al. [2007] também propuseram um método *branch-and-cut* para resolver o p-PLII com um modelo de otimização linear binária, usando uma metaheurística de busca tabu para os valores de entrada no solver. Contudo, só conseguiram tempo satisfatório para instâncias de média dimensão, nas quais p corresponde a $1/4$ da dimensão total das instâncias. Chiang e Lin [2017] também propuseram algoritmo *branch-and-cut* para resolver o problema, mas, da mesma forma, também só obtiveram resultados satisfatórios para instâncias de pequena dimensão,com até 200 clientes e 200 locais candidatos para escolher p até 20 instalações. Drezner et al. [2018] desenvolveram um método exato de localização que leva em consideração a distância mínima entre uma instalação e um cliente. Os resultados, todavia, só são satisfatórios para instâncias com um número reduzido de instalações a serem abertas. Colmenar et al. [2016] resolveram o problema usando um método heurístico baseado em GRASP. O algoritmo atingiu bons resultados em tempo hábil para a maioria das instâncias, sendo, no entanto, facilmente superado em instâncias em que o número de p instalações é reduzido e a quantidade de clientes e locais é elevada. Deve-se ressaltar que esta é uma realidade em problemas envolvendo instalações indesejadas, pois, geralmente, para esse tipo de problema, deseja-se localizar um número consideravelmente inferior de instalações em relação ao universo de busca. Drezner et al. [2019] desenvolveram um algoritmo baseado no diagrama de Voronoy e compararam com métodos exatos, usando o CPLEX para resolver um problema de localização de facilidades no plano.

Em um trabalho mais recente, Colmenar et al. [2018] abordam o problema multiobjetivo, no qual além de considerar que as instalações fiquem o mais afastado possível dos clientes, levam em consideração que as instalações devem se localizar dispersas entre si. Lin e Guan [2018] desenvolveram um algoritmo baseado na metaheurística PSO para o problema p-PLII e obtiveram resultados equivalentes aos anteriores apresentados na literatura. Herrán et al. [2018] propuseram um algoritmo baseado em VNS para resolver o problema em um conjunto de instâncias distintas e demonstraram, através de análises estatísticas, que o VNS é superior no estado da arte, sendo que as principais melhorias foram o tempo computacional e os resultados foram equivalentes aos anteriores encontrados na literatura.

3. Descrição do Problema

Esse artigo trata o p-PLII, que consiste em selecionar um subconjunto de p instalações de um determinado conjunto J de possíveis localizações, de forma que a soma da distância de todos os clientes até a instalação mais próxima seja maximizada. Considere C o conjunto de clientes, de modo que $|C| = n$ é a quantidade de clientes, $|J| = m$ a quantidade de instalações candidatas e seja S o conjunto de instalações abertas. A distância entre o cliente $i \in C$ e a instalação candidata $j \in J$ é representada por d_{ij}.

Um modelo matemático compacto, apresentado por Colmenar et al. [2016], para representar o problema é dado por:

$$\max \quad \sum_{i \in C} \min\{d_{ij} : j \in S\} \tag{1}$$

$$\text{Sujeito a:} \quad S \subseteq J, \ \ |S| = p \tag{2}$$

A função (1) representa a função objetivo do problema, que busca maximizar o somatório das distâncias mínimas entre todos os clientes até as facilidades pertencentes ao conjunto S das instalações abertas. As restrições (2) indicam que o conjunto S das instalações abertas deve estar contido no conjunto J das instalações candidatas e ter cardinalidade igual a p, garantindo que apenas sejam abertas as instalações necessárias.

Tamir [1991] demonstra que o p-PLII tratado é um problema NP-Difícil.

4. Metodologia

Nessa seção, apresenta-se como a solução foi representada, quais foram os movimentos utilizados, a função de avaliação utilizada, como é realizada a geração da solução inicial e, ao final, descreve-se o pseudocódigo do algoritmo completo implementado.

4.1. Representação da Solução

A solução x é representada através de um vetor binário, em que o valor 1 em uma dada posição representa uma instalação ativa e o valor 0 representa uma instalação inativa. O vetor de solução sempre terá dimensão m, ou seja, a quantidade de instalações candidatas.

Um exemplo para uma solução x com $p = 3$ instalações abertas e $m = 6$ instalações candidatas é dado por:

$$x = [0, 0, 1, 0, 1, 1]$$

em que as posições $i = 3$, $i = 5$ e $i = 6$ representam as instalações abertas e as demais as instalações inativas.

4.2. Movimentos de Exploração

O movimento de exploração do espaço de busca de soluções é dado pela vizinhança de troca. A troca é feita dois a dois, fechando uma instalação aberta, substituindo um 1 por um 0, e abrindo uma que esteja fechada, alterando um 0 por um 1.

Por exemplo, considere a solução x^1, em a instalação $i = 1$ está fechada e a instalação $i = 3$ aberta:

$$x^1 = [0, 0, 1, 0, 1, 1]$$

Então, após uma troca, a instalação $i = 1$ tornou-se aberta e a instalação $i = 3$ tornou-se fechada.

$$x^2 = [1, 0, 0, 0, 1, 1]$$

4.3. Função de Avaliação

A função avaliação está apresentada pela própria função objetivo do problema, ou seja:

$$f_{ava} = \sum_{i \in C} \min\{d_{ij} : j \in S\} \tag{3}$$

Nesta expressão, S representa o conjunto de instalações abertas, associado a um vetor x binário de solução, conforme descrito na Seção 4.1. Para cada cliente é verificada a distância à facilidade aberta mais próxima. Esse valor é somado para todos os clientes para se obter a soma das distâncias.

Algoritmo 1: Fase construtiva do GRASP adaptada ao problema p-PLII.

início
  para *w=1* **até** *numConstr* **faça**
    $S \leftarrow \emptyset$, $sum_i \leftarrow 0$
    para $j = 1$ **até** *m* **faça**
      $sum_j \leftarrow \sum_{i=1}^{n} d_{ij}$
    fim
    $F\_grasp = \min \sum_{i=1}^{n} d_{ij} + \alpha(\max \sum_{i=1}^{n} d_{ij} - \min \sum_{i=1}^{n} d_{ij})$
    Construa a LRC a partir de F_grasp
    para $iter = 1$ **até** p **faça**
      $j \leftarrow$ Elemento selecionado aleatoriamente da LRC
      $LRC \leftarrow LRC \setminus \{j\}$
      $S \leftarrow S \cup \{j\}$ solução aleatória em LRC
    fim
    Avalie S
    Atualize S^* caso necessário.
  fim
fim
retorna S^*

4.4. Geração da Solução Inicial

A solução inicial é gerada usando a fase construtiva da metaheurística GRASP, adaptada ao problema p-PLII. Inicialmente, as instalações são ordenadas, segundo a soma das distâncias de cada cliente para a instalação candidata. O conjunto ordenado de todas as possíveis instalações torna-se a Lista de Candidatos (LC). Em seguida, constrói-se a Lista Restrita de Candidatos (LRC) a partir da expressão:

$$F\_grasp = \min \sum_{i \in C} \{d_{ij} \ : \ j \in J\} + \alpha \left(\max \sum_{i \in C} \{d_{ij} \ : \ j \in J\} - \min \sum_{i \in C} \{d_{ij} \ : \ j \in J\} \right)$$

O valor F_grasp pode ser entendido como um ponto de corte na LC, que separa as instalações com valores superiores à F_grasp, que compõem a LRC, e as instalações com valores inferiores à F_grasp, que não entram na LRC. Observe, portanto, que a LRC, neste caso, não possui uma dimensão definida previamente, ou seja, uma cardinalidade previamente definida. O que a define é o valor do parâmetro α, pois, para α mais próximo de 1, a lista se torna mais restrita; para α quanto mais próximo de 0, menos restrita e, portanto, maior número de elementos. Ou, posto de outra forma, α=1 representa uma solução gulosa e α=0 uma solução completamente aleatória. Para compor a solução, deve-se sortear uma instalação, dentre os elementos da LRC, até preencher as p instalações necessárias.

Esta implementação está representada no Algoritmo 1. São geradas num_Constr construções e manteve-se apenas a melhor solução encontrada.

4.5. Algoritmo Híbrido GRASP-ILS

Para encontrar as melhores soluções para o p-PLII, propõe-se um Algoritmo Híbrido GRASP-ILS. Na primeira fase, o algoritmo utiliza a fase construtiva do GRASP, conforme mostrado no Algoritmo 1.

Algoritmo 2: Metaheurística ILS.

1 $f(S) \leftarrow S_{melhor\ GRASP}$
2 $Nivel\_Perturba \leftarrow 1$
3 **enquanto** $Nivel\_Perturba < Perturba\_Max$ **faça**
4   Aplica a busca local
5   **se** $f(S)\_Busca > f(S)$ **então**
6         $S \leftarrow S\_Busca$
7         $Nivel\_Perturba \leftarrow 1$
8   **fim**
9   **senão**
10   **fim**
11   Incrementa nível de perturbação
12 **fim**
13 **retorna** $S*$

Já na segunda fase do algoritmo é realizada a busca local por meio da metaheurística ILS, usando, como técnica de busca local, a heurística de primeira solução de melhora. Ou seja, o algoritmo realiza os movimentos de troca na vizinhança da solução corrente até que a primeira melhoria na função objetivo ocorra. Quando o algoritmo fica preso em um ótimo local é realizada uma perturbação que vai sendo incrementada em diferentes níveis, conforme a dificuldade de encontrar outro ótimo local, melhor que o anterior. O nível de perturbação é igual à quantidade de trocas realizadas pelo algoritmo.

A metaheurística ILS é apresentada no Algoritmo 2. Este algoritmo pode ser descrita conforme a seguir. Na linha 1, o algoritmo recebe a melhor solução construída pelo Algoritmo 1; na linha 2 o nível de perturbação recebe seu valor inicial igual a 1, que corresponde somente a uma troca. Na linha 3 inicia-se o *loop*, que limita o número de perturbações. Após isso, na linha 4, é feita a aplicação da heurística de busca local. O procedimento de verificação se a solução é melhor é feito na linha 5. Se a solução for melhor, a solução é armazenada e o nível de perturbação retorna para 1, conforme descrito nas linhas 6 e 7. Se a solução não melhorar, o nível de perturbação é incrementado e o procedimento é realizado novamente devido ao *loop*, conforme descrito entre as linhas 9 e 11. Quando o algoritmo alcança o nível de perturbação máximo, o *loop* encerra-se e o algoritmo retorna a melhor solução encontrada, conforme mostrado na linha 13.

O algoritmo híbrido GRASP-ILS constrói uma solução através da fase construtiva do GRASP e volta procurando melhores soluções causando perturbações para escapar de ótimos locais. O algoritmo se torna mais aleatório a cada nível de perturbação que não encontra uma solução melhor. Quando o algoritmo encontra uma solução melhor, ele retorna ao primeiro nível de perturbação. O procedimento completo do algoritmo unindo as duas fases pode ser reapresentado da seguinte maneira:

Algoritmo 3: Algoritmo híbrido GRASP-ILS

para $w = 1$ **até** num_Constr **faça**
  $S \leftarrow \emptyset$, $sum_i \leftarrow 0$
  para $j = 1$ **até** m **faça**
    $sum_j \leftarrow \sum_{i=1}^{n} d_{ij}$
  fim
  $F\_grasp = menor(sum_j) + \alpha(\ maior(sum_j) - menor(sum_j))$
  Construa a LRC a partir de F_grasp
  para $iter = 1$ **até** p **faça**
    $j \leftarrow$ Elemento selecionado aleatoriamente da LRC
    $LRC \leftarrow LRC \setminus \{j\}$
    $S \leftarrow S \cup \{j\}$ solução aleatória em LRC
  fim
  Avalie S
  Atualize S_{melhor} caso necessário.
fim
$S* \leftarrow S_{melhor}$
$f(S) \leftarrow S*$
$Nivel\_Perturba \leftarrow 1$
enquanto $Nivel\_Perturba < Perturba\_Max$ **faça**
  Aplica a busca local
  se $f(S)\_Busca > f(S)$ **então**
    $S \leftarrow S\_Busca$
    $Nivel\_Perturba \leftarrow 1$
  fim
  senão
    Incrementa nível de perturbação
  fim
fim
retorna $S*$

5. Resultados Computacionais

Para a realização dos experimentos foram utilizadas instâncias disponibilizadas por Colmenar et al. [2016]. O conjunto de instâncias executadas compreendem matrizes de 200×200 com p instalações de 25, 50 e 100. Foram executadas 4 instâncias para cada valor de p, totalizando 24 instâncias. O algoritmo proposto foi desenvolvido na linguagem `C++`, utilizando-se o IDE Code::Blocks. Todos os experimentos foram executados em um notebook Core i5-4200U CPU @ 1.60 GHz com 8 GB de memória RAM e sistema operacional Windows 7. Os parâmetros utilizados pelo algoritmo, escolhidos de maneira empírica, foram $\alpha = 0,9$ e nível máximo de perturbação $Perturba\_MAX = 10$. Foram geradas $num\_Constr = 10.000$ soluções iniciais na fase construtiva do GRASP.

A Tabela 1 apresenta a comparação entre os resultados obtidos pela aplicação do procedimento computacional GRASP-ILS, mostrado no Algoritmo 3 aqui proposto, com as melhores soluções encontradas na literatura para estas mesmas instâncias, conforme reportado por Herrán et al. [2018], para os algoritmos GRASP [Colmenar et al., 2016], B&C+XTS [Belotti et al., 2007], XTS [Belotti et al., 2007] e VNS [Herrán et al., 2018]. Para fins de análise, são calculados os desvios (*gap*) entre os resultados encontrados por este procedimento e o melhor resultado mostrado em Herrán et al. [2018]. O cálculo dos valores de *gap* é feito da seguinte forma:

$$gap_{GI}\% = \left(\frac{f(GI) - f(Alg\_Melhor)}{f(Alg\_Melhor)}\right) \times 100.$$

em que $f(Alg\_Melhor)$ representa a melhor solução gerada pelo melhor algoritmo Alg_Melhor da literatura, conforme Herrán et al. [2018], e $f(GI)$ representa o valor da solução obtida pelo procedimento aqui proposto. Esta análise é feita a partir de 10 execuções do algoritmo GRASP-ILS, realizadas a partir de 10 diferentes sementes, sendo calculadas a média e o desvio padrão entre as soluções para cada instância.

Tabela 1: Comparação entre os resultados encontrados pela aplicação do Algoritmo GRASP-ILS proposto às instâncias da literatura e os algoritmos GRASP [Colmenar et al., 2016], B&C+XTS [Belotti et al., 2007], XTS [Belotti et al., 2007] e VNS [Herrán et al., 2018].

Instância	$f(Alg\_Melhor)$	$f(GI)$	$\bar{\mu}$	σ^2	$gap_{Alg\_Melhor}\%$	$gap_{GI}\%$	Tempo (s)
pmed17-p100.A.txt	**4054**	3992	3979	16,1	0,00	1,53	428
pmed17-p25.A.txt	**7317**	6905	6901	7,38	0,00	5,63	294
pmed17-p50.A.txt	5411	**5563**	5527	21,41	2,73	0,00	367
pmed18-p100.A.txt	**4220**	4119	4107	6,74	0,00	2,39	681
pmed18-p25.A.txt	7432	**7662**	7662	0	3,00	0,00	131
pmed18-p50.A.txt	5746	**5852**	5852	0	1,81	0,00	229
pmed19-p100.A.txt	**4033**	4015	3972	26,93	0,00	0,45	600
pmed19-p25.A.txt	**7020**	6816	6812	12,36	0,00	2,91	168
pmed19-p50.A.txt	5387	**5423**	5405	30,03	0,66	0,00	284
pmed20-p100.A.txt	4063	**4066**	4026	13,25	0,07	0,00	533
pmed20-p25.A.txt	**7648**	7349	7349	0	0,00	3,91	132
pmed20-p50.A.txt	**5872**	5665	5643	0	0,00	3,53	524
pmed17-p100.B.txt	3992	**4054**	4054	0	1,53	0,00	375
pmed17-p25.B.txt	6905	**7317**	7316	1,2	5,63	0,00	312
pmed17-p50.B.txt	**5563**	5411	5409	5,2	0,00	2,73	276
pmed18-p100.B.txt	4122	**4218**	4204	0	2,28	0,00	377
pmed18-p25.B.txt	**7662**	7432	7432	0	0,00	3,00	305
pmed18-p50.B.txt	**5852**	5746	5723	24,42	0,00	1,81	688
pmed19-p100.B.txt	4016	**4033**	3994	26,93	0,42	0,00	700
pmed19-p25.B.txt	6816	**7020**	7020	0	2,91	0,00	200
pmed19-p50.B.txt	**5423**	5377	5354	13	0,00	0,85	852
pmed20-p100.B.txt	**4067**	4062	4051	13,2	0,00	0,12	653
pmed20-p25.B.txt	7349	**7648**	7648	0	3,91	0,00	269
pmed20-p50.B.txt	5665	**5872**	5354	4,0	3,53	0,00	1060
		MÉDIA =			1,19	1,20	435

Analisando esta Tabela, verifica-se que o procedimento proposto encontrou bons resultados em relação aos principais algoritmos da literatura, tendo um *gap* médio de $1,20\%$ em relação às melhores soluções encontradas pela literatura. Em boa parte das instâncias com p instalações igual a 25 e 50, o procedimento superou os resultados, o que é positivo, visto que, para um problema dessa dimensão, geralmente tem-se um espaço de busca muito maior e as instalações costumam ser reduzidas. Este fato é corroborado, por exemplo, pela análise do cenário atual brasileiro, em que existem 1.700 aterros sanitários e 1.478 prisões, para um total de 5570 municípios existentes no Brasil IBGE [2018]. Isso indica que a relação mais próxima à realidade são instâncias em que o número de p instalações segue uma razão de aproximadamente 25% em relação ao tamanho da instância. Assim, os resultados mais relevantes dessa heurística são os que tratam $p = 50$.

6. Conclusão e Trabalhos Futuros

Este trabalho teve seu foco no problema de localização de instalações indesejadas p-PLII, que consiste em localizar p instalações de modo a maximizar o somatório das distâncias mínimas, de cada cliente às instalações abertas.

Foi proposto um algoritmo heurístico híbrido, que combina técnicas das metaheurísticas GRASP e ILS. A primeira etapa corresponde à fase construtiva do GRASP e, à segunda fase, corresponde a busca local, usando a técnica de perturbações do algoritmo ILS. O algoritmo foi testado em um conjunto de instâncias da literatura e os resultados foram comparados com os melhores encontrados na literatura para o problema.

Os resultados obtidos indicam que o algoritmo híbrido GRASP-ILS mostrou-se tão eficiente quanto os melhores algoritmos da literatura, pois superou os algoritmos da literatura em 12 instâncias e foi pior em 12 instâncias. A comparação percentual dos algoritmos também indica equivalência, pois o *gap* percentual teve diferença de $0,01\%$ entre os algoritmos.

Além disso, neste artigo discutiu-se a relevância de testar instâncias com valores de p muito altos (valores superiores a 25% em relação ao total de instalações possíveis), visto que, na maioria dos casos de localização de instalações indesejadas, a relação é de até 25%. Para validar essa observação, comparou-se os dados referentes a quantidades de presídios e quantidades de aterros sanitários no Brasil em relação a quantidades de municípios existentes no país.

Como trabalhos futuros, apontam-se os seguintes: (i) implementar a solução do problema usando métodos exatos e verificar o limite de dimensão de instâncias viável para ser solucionado via essa técnica; (ii) executar a calibração dos parâmetros da metaheurística híbrida GRASP-ILS; (iii) explorar métodos híbridos entre método exato usando o CPLEX e metaheurísticas, com inicialização por meio de solução oriunda desta metaheurística.

Agradecimentos

Os autores agradecem a CAPES, ao CNPq, à FAPEMIG e ao CEFET-MG pelo apoio ao desenvolvimento desta pesquisa.

Referências

Batta, R. e Chiu, S. S. (1988). Optimal obnoxious paths on a network: transportation of hazardous materials. *Operations Research*, 36(1):84–92.

Belotti, P., Labbé, M., Maffioli, F., e Ndiaye, M. M. (2007). A branch-and-cut method for the obnoxious p-median problem. *4OR*, 5(4):299–314.

Cappanera, P., Gallo, G., e Maffioli, F. (2003). Discrete facility location and routing of obnoxious activities. *Discrete Applied Mathematics*, 133(1-3):3–28.

Chiang, Y.-I. e Lin, C.-C. (2017). Compact model for the obnoxious p-median problem. *American Journal of Operations Research*, 7(06):348.

Colmenar, J. M., Greistorfer, P., Martí, R., e Duarte, A. (2016). Advanced greedy randomized adaptive search procedure for the obnoxious p-median problem. *European Journal of Operational Research*, 252(2):432–442.

Colmenar, J. M., Martí, R., e Duarte, A. (2018). Multi-objective memetic optimization for the bi-objective obnoxious p-median problem. *Knowledge-Based Systems*, 144:88–101.

Drezner, T., Drezner, Z., e Schöbel, A. (2018). The Weber obnoxious facility location model: A Big Arc Small Arc approach. *Computers & Operations Research*, 98:240–250.

Drezner, Z., Kalczynski, P., e Salhi, S. (2019). The planar multiple obnoxious facilities location problem: A Voronoi based heuristic. *Omega*, 87:105–116.

Fernández, J., Fernández, P., e Pelegrın, B. (2000). A continuous location model for siting a non-noxious undesirable facility within a geographical region. *European Journal of Operational Research*, 121(2):259–274.

Goldman, A. J. e Dearing, P. M. (1975). Concepts of optimal location for partially noxious facilities. *ORSA Bulletin*, 23(1).

Gopalan, R., Kolluri, K. S., Batta, R., e Karwan, M. H. (1990). Modeling equity of risk in the transportation of hazardous materials. *Operations Research*, 38(6):961–973.

Herrán, A., Colmenar, J. M., Martí, R., e Duarte, A. (2018). A parallel variable neighborhood search approach for the obnoxious p-median problem. *International Transactions in Operational Research*.

Hosseini, S. e Esfahani, A. M. (2009). Obnoxious facility location. In Zanjirani Farahani, R. e Hekmatfar, M., editors, *Facility Location: Concepts, Models, Algorithms and Case Studies*, p. 315–345. Physica-Verlag HD.

IBGE (2018). Instituto Brasileiro de Geografia e Estatística (Tabelas da Divisão Territorial Brasileira 2018). URL `https://www.ibge.gov.br/geociencias/organizacao-do-territorio/divisao-regional/23701-divisao-territorial-brasileira.html?=&t=acesso-ao-produto`.

Labbé, M., Maffioli, F., Ndiaye, M., e Belotti, P. (2001). Obnoxious p-median problems: valid inequalities and a branch-and-cut approach. *The OR Peripatetic Post-Graduate Programme*, p. 26–29.

Lin, G. e Guan, J. (2018). A hybrid binary particle swarm optimization for the obnoxious p-median problem. *Information Sciences*, 425:1–17.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2003). Iterated local search. In *Handbook of metaheuristics*, p. 320–353. Springer.

Melachrinoudis, E., Min, H., e Wu, X. (1995). A multiobjective model for the dynamic location of landfills. *Location Science*, 3(3):143–166.

Rakas, J., Teodorović, D., e Kim, T. (2004). Multi-objective modeling for determining location of undesirable facilities. *Transportation Research Part D: Transport and Environment*, 9(2):125–138.

Resende, M. G. e Ribeiro, C. (1998). Greedy randomized adaptive search procedures (grasp). *AT&T Labs Research Technical Report*, 98(1):1–11.

Tamir, A. (1991). Obnoxious facility location on graphs. *SIAM Journal on Discrete Mathematics*, 4(4):550–567.