EAComp
Encontro Acadêmico de Computação

VIII Encontro Acadêmico de Computação - UFMA
Copyright (c) EAComp. ISSN: 2594-486X

Meta-heurística GRASP Aplicada ao Problema de Alocação de Hubs Capacitados: Estudo de Caso com Base de Dados da Empresa LOGGI

Nathasha Pinto[1], **Glaubos Clímaco**[2]

[1]Universidade Federal do Maranhão (UFMA)
Av. dos Portugueses, 1966 - Vila Bacanga, Sao Luís - MA, 65080-805

[2]Departamento de Informática (DEINF)
Universidade Federal do Maranhão (UFMA) – São Luís - MA – Brasil

nathasha.pinto@discente.ufma.br, francisco.glaubos@ufma.br

Resumo. *Em setores que exigem uma logística exemplar, como por exemplo em uma empresa de entregas, problemas de otimização são comuns. O fluxo de entrada e saída de objetos é muito grande e por isso é necessário se utilizar de métodos heurísticos que minimizam os custos de transporte. O presente artigo descreve um estudo de caso utilizando a meta-heurística GRASP no problema de localização de hubs capacitados, buscando minimizar o custo total de deslocamento. O problema foi adaptado pensando em seu uso real, no qual as instâncias utilizadas para os testes computacionais foram geradas a partir de uma base de dados real.*
Palavras-chave*: GRASP, meta-heurística, PpHC, Loggi.*

Abstract. *In industries that require exemplary logistics, such as a delivery company, optimization problems are common. The inflow and outflow of objects is very large and that is why it is necessary to use heuristic methods that minimize transport costs. This article describes a case study using the GRASP meta-heuristic in the problem of locating capable hubs, seeking to minimize the total cost of travel. The problem was adapted thinking about its real use, the instances used for the computational tests came from a database.*
Key-words*: GRASP, meta-heuristic, HLPs.*

1. Introdução

Problemas de otimização consistem em achar a melhor combinação dentre um conjunto de variáveis para maximizar ou minimizar uma função, geralmente chamada de função objetivo ou função custo [Becceneri 2008]. Esses problemas de otimização no contexto de uma rede de entregas são frequentes, uma vez que o fluxo de entrada e saída é muito grande. Há diversos métodos exatos que em teoria podem ser aplicados para a resolução de tais problemas, mas na prática, ao serem expostos a instâncias de grande porte se tornam complexos e inviáveis. Para isso foram criadas meta-heurísticas, que são estruturas algorítmicas independentes que fornecem um conjunto de diretrizes ou estratégias para desenvolver algoritmos de otimização e que podem gerar soluções viáveis para problemas reais [Glover e Kochenberger 2006].

Quando a transmissão de dados entre os pontos não se dá de forma direta, essa rede é chamada de *hub-and-spoke* [Aykin 1994], e os problemas existentes nessa área se dão

justamente na comunicação entre os pontos não-hubs e os hubs, pois é necessário avaliar qual o melhor meio para que essa transmissão ocorra, considerando o custo mínimo. A partir disso surge o problema de alocação de hubs (PpH), problema que está inserido na classe NP-difícil [Garey e Johnson 1979], cujo o objetivo é encontrar a melhor localização dos hubs e alocar pontos não-hubs à eles, minimizando uma função objetivo que descreve o fluxo intercambiado e seu custo.

Sendo assim, neste trabalho será abordado um estudo de caso sobre o problema de alocação de p-hubs capacitado (PpHC). Esse problema consiste em uma variante do problema formulado por [O'kelly 1986] e tem como parâmetros que diferem do principal problema: um número fixo de p hubs, a capacidade máxima dos hubs selecionados, a restrição dos pontos (o ponto só pode se ligar a um hub), e o custo de configuração associado.

No intuito de resolver um problema real, o presente estudo utiliza instâncias reais advindas de uma base de dados fornecida pela LOGGI, uma empresa de entregas. E nessas instâncias será aplicada uma heurística baseada em GRASP (Greedy Randomized Adaptive Search Procedures) [Feo e Resende 1995]. Com o GRASP será obtida uma solução em duas fases; a primeira será a aplicação de um algoritmo guloso para a construção de uma solução, e em seguida será feita uma busca local nessa solução, a fim de aprimorá-la.

2. O PpHC na literatura

O problema de alocação de hubs não é um problema recente tendo sua primeira solução proposta por [O'kelly 1987], a partir de um modelo de função objetivo quadrática não convexa e restrições lineares. A primeira formulação totalmente linear foi desenvolvida por [Campbell 1992], permitindo o uso de métodos padrões de programação linear para a resolução do problema.

Em trabalhos mais recentes como o de [Lüer-Villagra et al. 2019], é sugerido o uso de uma combinação de algoritmo genético com outro modelo matemático aplicado, fazendo uso de até 200 pontos em seus testes. A conclusão obtida é que a meta-heurística aplicada permite a resolução de pequenas e médias instâncias de forma rápida e satisfatória, mas quando aplicada a grandes instâncias, se torna custosa.

Para a resolução do PpH, a maioria dos pesquisas fazem uso de heurísticas híbridas, como mostra [Sangsawang e Chanta 2020] que faz uso de VNS (*Variable Neighborhood Search*) juntamente com TS (*Tabu Search*) que podem ser aplicados em cenários reais. Já os autores [Alkaabneh et al. 2019] fazem uma comparação entre o uso de GRASP e de uma relaxação Lagrangiana, mostrando que ambas produzem resultados de alta qualidade, com destaque para o GRASP que foi capaz de gerar soluções ótimas em quase 90% dos testes.

Um estudo realizado por [de Carvalho 2017] apresenta uma proposta de solução utilizando uma hibridização de meta-heurísticas para o problema que será analisado no atual trabalho, o PpHC. O autor traz um método nomeado ILS-RVND-PR que é um fusão das técnicas *Iterated Local Search* [Lourenço et al. 2010], *Random Variable Neighborhood Descent* [Souza et al. 2010] e *Path-Relinking* [Reeves 1997], que mostra que a aplicação foi capaz de encontrar soluções de boa qualidade em pouco tempo.

Copyright (c) EAComp. ISSN: 2594-486X

Como foi observado, muitos autores propõem abordagens distintas buscando encontrar a melhor solução solução mais razoável para o problema. Deste modo os autores do presente trabalho propõem o uso da meta-heurística GRASP, para resolver o PpHC a partir de instâncias inéditas baseadas em dados reais, a fim de verificar a sua viabilidade de aplicação na indústria.

3. Método de solução

Neste trabalho, propomos uma heurística baseada na meta-heurística GRASP proposta por [Feo e Resende 1995], que é um método iterativo constituído por duas fases: a de construção e a de busca local.

Na fase de construção, elementos que podem compor uma solução são inseridos em uma lista de candidatos (LC), que então é ordenada de acordo com a contribuição do elemento na função objetivo. Em seguida, os melhores elementos de LC são escolhidos para compor uma lista de candidatos restrita (LCR). Por fim, escolhe-se, aleatoriamente, um elemento de LRC para compor a solução. Esse processo é repetido até que uma solução viável para o problema seja obtida.

De acordo com [Feo e Resende 1995], não há garantias que a solução construída seja localmente ótima. Portanto, a partir da solução construída, é realizado um procedimento de refinamento conhecido como Busca Local.

O Algoritmo 1 apresenta um pseudo-código do método GRASP, no qual *Seed* representa uma semente para as componentes aleatórias dentro do método, e *MaxIterations* é a quantidade de iterações que o GRASP irá executar. As listas LC e LCR serão geradas com a função *GreedyRandomizedConstruction*, e delas será gerada uma solução parcial. Já a busca local é feita na função *LocalSolution*, para refinar a solução.

Algorithm 1 Pseudoalgoritmo GRASP

```
function GRASP(MaxIterations, Seed)
    for k = 0, ..., MaxIterations do
        Solution ← GreedyRandomizedConstruction(Seed)
        Solution ← LocalSolution(Solution)
        UpdateSolution(Solution, BestSolution)
    end for
     return BestSolution
end function
```

3.1. Fase de construção

A fase de construção desenvolvida neste trabalho, difere da construção tradicionalmente feita com o GRASP. Na nossa proposta, inicialmente são selecionados p pontos da rede para serem hubs, e em seguida, são realizadas ligações de pontos não-hubs (V) a hubs (H), de forma que minimize o custo da solução, que se define pela soma total do custo de todas as ligações entre hubs e não-hubs. Por fim, essa fase retorna uma solução $S(H, V)$ contendo associações entre hubs e não-hubs. O Algoritmo 2 mostra com mais detalhes como são realizadas essas associações.

Algorithm 2 Algoritmo que liga os vértices aos hubs

A partir de V escolher aleatoriamente p hubs para compor H
$\bar{V} \leftarrow V$
while $\bar{V} \neq \emptyset$ **do**
  escolher o $v \in \bar{V}$ mais próximo a um hub $h \in H$
  $v.hub = h$ ▷ Associando v ao hub h, caso a capacidade de h seja respeitada
  Atualizar a capacidade do hub h
  $\bar{V} \leftarrow \bar{V} \setminus v$
end while
return S

3.2. Fase da busca local

O pseudo-código de busca local aplicado a partir de uma solução construída S é representado pelo Algoritmo 3. Para cada vértice não-hub é feita uma tentativa de troca de associação com um diferente hub, e caso essa troca resulte em uma solução de menor custo, essa nova associação é estabelecida, e as capacidades dos hubs envolvidos na troca são atualizadas.

Algorithm 3 Busca_Local(S)

melhor_custo = custo(S)
melhor_solucao = S
$S' = S$
for cada $v \in V'$ **do**
  for cada $h \in H'$ **do**
    if $h = v.hub$ **then**
      Continue ▷ Nada é feito, e o próximo v é considerado
    end if
    $v.hub = h$ ▷ associando vértice v a um outro hub
    Atualizar a solução S' ▷ atualiza a solução caso a troca seja viável
    if custo(S') $<$ custo(S) **then**
      melhor_custo = custo(S')
      melhor_solucao = S'
    else
      desfazer a associação
    end if
  end for
end for
return melhor_solucao

4. Experimentos Computacionais

Nesta sessão serão apresentados os resultados obtidos com a aplicação do método proposto, e os seus resultados serão comparados com resultados obtidos a partir do algoritmo guloso 2. Neste trabalho, supomos que em situações reais de tomada de decisão acerca do PpHC, geralmente utiliza-se estratégias gulosas para a alocação de hubs.

Todos os códigos-fonte foram implementados na linguagem de programação Python3, em um computador pessoal Intel® Core™ i3 CPU@2.30GHz com 8GB RAM, utilizando o sistema operacional Linux.

4.1. Instâncias do estudo de caso

Para a realização dos experimentos computacionais, considerou-se um estudo de caso que se constitui de instâncias geradas a partir de uma base de dados real da empresa Loggi [Loggi 2021]. A base de dados utilizada contém localizações reais distribuídas em três estados: Pará, Distrito Federal e Rio de Janeiro. Foram extraídos dessa base o número de hubs que cada estado contém, as instâncias que indicavam a coordenada geográfica, e a demanda de cada ponto (ou cliente). As instâncias geradas variam de tamanho, tendo a menor 50 pontos e a maior 200, parâmetro definido com bases nos trabalhos anteriormente analisados [Sangsawang e Chanta 2020, de Carvalho 2017, Lüer-Villagra et al. 2019].

Após essa extração foi necessário descobrir as distâncias entre todos os pontos, uma vez que o método de solução proposto necessita dessa informação. Para descobrir a localização real dos pontos e a distância entre eles, utilizou-se a Distance Matrix API fornecida pela Google. Uma vez que ela faz essa verificação, ela retorna o valor da distância em metros, incluindo do ponto para ele mesmo. Em seguida, as distâncias foram convertidas para valores em quilômetros, e visto que todos os valores são comparados inclusive do ponto para ele mesmo, foi necessário modificar esse valor para um valor alto (10000), eliminando assim a possibilidade de que o algoritmo considere um ponto mesmo como opção de ligação para si mesmo.

Essas distâncias juntamente com a demanda de cada nó, foram escritas em um arquivo `.txt`, organizados da forma como mostra o exemplo na (Figura 1), contendo a quantidade de vértices, a demanda de cada um e por fim a matriz distância criada pela API.

```
5 //Quantidade de nós//
9
2
8
8
1
1000.0          18.4          18.6          17.3          22.7
18.4            1000.0        0.2           7.7           40.7
18.6            0.2           1000.0        7.9           40.9
17.3            7.7           7.9           1000.0        33.5
23.3            41.2          41.4          34.1          1000.0
```

Figura 1. Arquivo TXT ultilizado

Os hubs considerados no problema possuem uma capacidade de atendimento de demandas, porém tal capacidade não é fornecida pela base de dados da LOGGI, então usou-se uma fórmula matemática para atender com menor custo (sem desperdício de espaço), o tamanho de cada hub. A Fórmula 1 se dá pela soma da capacidade de todos os pontos (c) dividida pela multiplicação da quantidade de hubs (h) e de um valor x que representa a capacidade que será definida ao hub. O resultado da operação é a taxa de dificuldade (tx) que o algoritmo pode obter. Essa taxa de dificuldade é dada com valores entre 0 e 1, e quão mais próximo de 1 o valor obtido pela fórmula for, mais difícil a resolução do problema será. Para o calculo de todas as capacidades das instâncias utilizadas, assumiu-se $tx = 0,9$.

Copyright (c) EAComp. ISSN: 2594-486X

$$\frac{c}{h * x} = tx \tag{1}$$

As capacidades dos hub foram adicionadas manualmente ao algoritmo, alterando-se de acordo com a variação dos tamanhos das instâncias: 50, 100, 150, 200. A Figura 2 ilustra uma solução para uma instância do PpHC, na qual são mostrados os hubs e os pontos ligados a eles juntamente com o custo total.

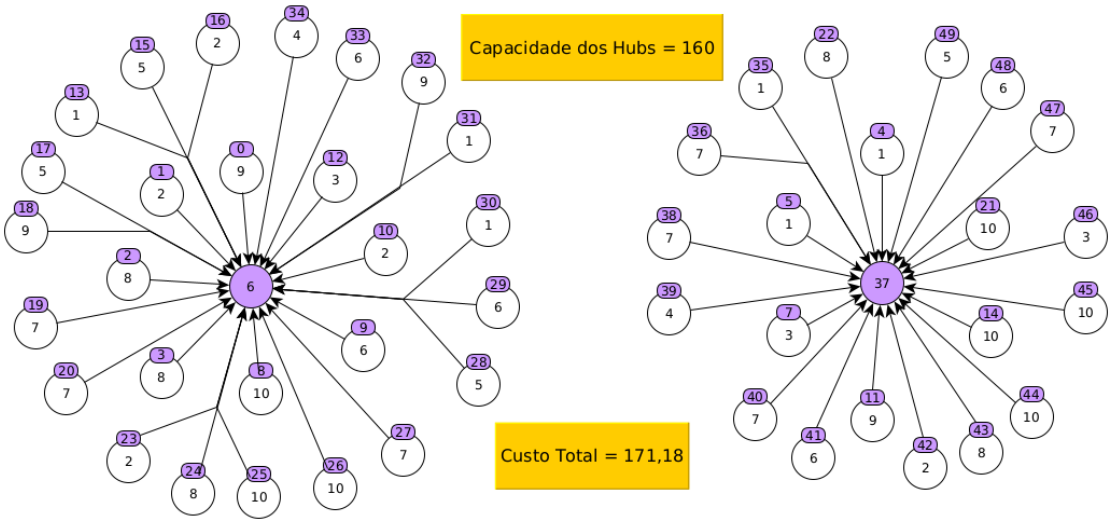

Figura 2. Solução gerada a partir da instância de 50 pontos do estado do Pará

4.2. Resultados numéricos

A Tabela 1 mostra os resultados obtidos dos testes realizados com GRASP e um algoritmo guloso, exceto informações de quais hubs foram escolhidos na execução e quais pontos estão ligados a ele. Na primeira coluna é exibido o tamanho da instância (quantidade de pontos), e nas colunas seguintes, a quantidade de hubs, a média de custos, a média de tempo e o melhor custo dentre dez execuções do algoritmo GRASP. Nas colunas seis e sete, são exibidos o custo e o tempo utilizando pelo algoritmo guloso. Cada estado verificado contém quatro tamanhos diferentes de instâncias que variam entre 50 e 200. O tempo de execução é dado em segundos, e o custo em quilômetros, que se trata da soma de todas as distâncias percorridas entre cada hub e seus pontos associados.

Tam. Instância	Q. Hubs	Média Custos	Média Tempo	Melhor Custo	Tempo	Custo
PA		GRASP			ALGORITMO GULOSO	
50	2	183,18	0,02647	177,1	0,00587	183,0
100		382,31	0,12620	365,2	0,01257	381,4
150		914,78	0,27501	886,6	0,04531	898,5
200		1802,546	0,45853	1752,9	0,08646	1820,2
DF		GRASP			ALGORITMO GULOSO	
50	3	106,18	0,04905	40,5	0,00811	467,2
100		302,53	0,11353	119,8	0,01103	796,3
150		570,25	0,24586	360,5	0,04421	1149,2
200		857,81	0,42185	748,8	0,05904	1338,7
RJ		GRASP			ALGORITMO GULOSO	
50	7	5,24	0,01755	4,0	0,01093	48,9
100		10,35	0,06192	6,1	0,09646	123,7
150		25,65	0,14467	14,2	0,03378	260,1
200		42,86	0,23688	31,4	0,04868	506,9

Tabela 1. Tabela de resultados dos testes com GRASP e Algoritmo Guloso

Como verificado na Tabela 1, a quantidade de pontos (clientes) para atender influencia diretamente no custo total e na média de tempo gasto. Quanto mais cliente para atender, mais ligações são feitas entre pontos não-hubs e hubs.

Considerando os três estados, temos que o PA e DF apresentam os maiores custos, enquanto o RJ apresenta os menores custos. Como nas instâncias as distâncias entre dois pontos não costumam variar muito, atribuímos essa diferença de valor de solução à densidade da rede. As instâncias de RJ são mais densas que as demais, com mais rodovias e opções de ligação entre pontos não-hubs e hubs, o que pode causar também um aumento no tempo de processamento.

Analisando os resultados da meta-heurística e do algoritmo guloso pode-se notar que há grande diferença nos resultados. Em relação ao custo da solução obtida, o método GRASP foi superior em todos os casos, utilizando um tempo computacional similar ao do algoritmo guloso. Por exemplo, nos valores referentes ao Distrito Federal, ao compará-los na menor instância (50 pontos) há um decréscimo de mais de 90% no custo total ao utilizar o GRASP.

5. Conclusão

Problemas de otimização são cada vez mais comuns, ainda mais inseridos em uma rede de entregas, como a empresa Loggi. A entrada e saída de mercadorias tem um fluxo muito grande, havendo sempre que melhorar a logística do negócio. Sabe-se que buscar sempre os menores custos é uma forma de obter lucro e crescer a companhia, por conta disso nesse trabalho foi proposto o uso da meta-heurística GRASP adaptada ao problema de localização de hubs capacitados (PpHC), que visa minimizar o custo total do fluxo de mercadorias.

Utilizou-se uma base de dados da empresa com pontos reais, conferindo assim uma aplicação real do trabalho. Os resultados foram gerados a partir de testes em doze instâncias de tamanhos variados e apresentados em uma tabela mostrando o tempo médio,

o custo médio, e o melhor custo de cada execução separados por estado. O trabalho mostrou bons resultados para o problema, conseguindo atingir soluções satisfatórias quando comparadas a um algoritmo puramente guloso, em um tempo computacional interessante.

Como trabalhos futuros pretende-se buscar na literatura formas de melhorar o algoritmo tanto em relação ao tempo quanto ao custo, e para isso, técnicas como Mineração de Dados e Reconexão de Caminhos devem ser analisadas [Clímaco et al. 2019]. Além disso, pretende-se também desenvolver uma interface gráfica que facilite a utilização do método por meio de mapas interativos.

Referências

Alkaabneh, F., Diabat, A., e Elhedhli, S. (2019). A lagrangian heuristic and grasp for the hub-and-spoke network system with economies-of-scale and congestion. *Transportation Research Part C: Emerging Technologies*, 102:249–273.

Aykin, T. (1994). Lagrangian relaxation based approaches to capacitated hub-and-spoke network design problem. *European Journal of Operational Research*, 79(3):501–523.

Becceneri, J. C. (2008). Meta-heurísticas e otimização combinatória: Aplicações em problemas ambientais. *INPE, São José dos Campos*, pages 65–81.

Campbell, J. F. (1992). Location and allocation for distribution systems with transshipments and transportion economies of scale. *Annals of operations research*, 40(1):77–99.

Clímaco, G., Rosseti, I., Simonetti, L., e Guerine, M. (2019). Combining integer linear programming with a state-of-the-art heuristic for the 2-path network design problem. *International Transactions in Operational Research*, 26(2):615–641.

de Carvalho, R. (2017). Heurísticas paralelas aplicadas a problemas de alocação de concentradores. *Teses de Doutorado UFMG*.

Feo, T. A. e Resende, M. G. (1995). Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133.

Garey, M. R. e Johnson, D. S. (1979). *Computers and intractability*, volume 174. freeman San Francisco.

Glover, F. W. e Kochenberger, G. A. (2006). *Handbook of metaheuristics*, volume 57. Springer Science & Business Media.

Loggi (2021). loggibud: Loggi benchmark for urban deliveries. `https://github.com/loggi/loggibud`.

Lourenço, H. R., Martin, O. C., e Stützle, T. (2010). Iterated local search: Framework and applications. In *Handbook of Metaheuristics*, pages 363–397. Springer.

Lüer-Villagra, A., Eiselt, H. A., e Marianov, V. (2019). A single allocation p-hub median problem with general piecewise-linear costs in arcs. *Computers & Industrial Engineering*, 128:477–491.

O'kelly, M. E. (1986). The location of interacting hub facilities. *Transportation science*, 20(2):92–106.

 ISSN: 2594-486X

O'kelly, M. E. (1987). A quadratic integer program for the location of interacting hub facilities. *European journal of operational research*, 32(3):393–404.

Reeves, C. (1997). Modern heuristics techniques for combinatorial problems. *Nikkan Kogyo Shimbun*.

Sangsawang, O. e Chanta, S. (2020). Capacitated single-allocation hub location model for a flood relief distribution network. *Computational Intelligence*, 36(3):1320–1347.

Souza, M. J., Coelho, I. M., Ribas, S., Santos, H. G., e Merschmann, L. H. d. C. (2010). A hybrid heuristic algorithm for the open-pit-mining operational planning problem. *European Journal of Operational Research*, 207(2):1041–1051.