



20 e 21 de outubro
Instituto Nacional de Pesquisas Espaciais - INPE
São José dos Campos - SP

Abordagens meta-heurísticas para o problema de localização de concentradores com restrições de capacidade

Wesley G. Almeida¹, Edson L. F. Senne², Horacio H. Yanasse¹

¹ Instituto Nacional de Pesquisas Espaciais – INPE
Caixa Postal 515 – 12.227-010 – São José dos Campos – SP – Brazil

² Universidade Estadual Paulista – UNESP Faculdade de Engenharia – Campus de Guaratinguetá Caixa Postal 205 - 12516-410 Guaratinguetá, SP, Brasil
wesgomes2005@gmail.com, elfsenne@feg.unesp.br, horacio@lac.inpe.br

Resumo. Neste trabalho estuda-se o problema de localização de concentradores com restrições de capacidade. Diversas abordagens são apresentadas com objetivo de mostrar as eficiência e eficácia de cada método.

Abstract. In this paper it is studied the hub location problem with capacity constraints. Several approaches are presented aiming to show the efficiency and effectiveness of each method.

Palavras-chave: Meta-heurísticas; Simulated Annealing; VNS; Algoritmos Genéticos; Problema de Localização de Concentradores.

1. Introdução

O problema de localização de concentradores (PLC) tem com principal objetivo minimizar o custo total de transporte de alguma entidade em uma rede com n nós de demanda, dado o fluxo entre cada par de nós origem-destino. A solução do problema busca encontrar os nós que devem se tornar concentradores e a alocação dos demais nós a estes concentradores de forma que o custo total da rede seja minimizado.

Em alguns problemas definidos em redes, a comunicação entre os nós da rede não acontece de forma direta, mas por meio de nós especiais denominados concentradores. Isto ocorre com frequência, por exemplo, em redes de transporte e em redes de telecomunicação. Nestes casos, diz-se que a rede é do tipo *hub-and-spoke* (Aykin, 1994).

Diversas aplicações práticas em redes de transporte e de telecomunicações, como o transporte de pessoas, produtos, materiais ou de dados, ocorrem em redes deste tipo. A Figura 1 mostra um exemplo de rede do tipo *hub-and-spoke*, onde os concentradores são representados por quadrados e os demais nós (denominados *spokes* ou nós de demanda) são representados por círculos.

Existem diferentes versões do problema de localização de concentradores (*hubs*). Alguns casos podem apresentar restrições de capacidade (AYKIN, 1994), ou seja, um limitante no volume de informações que um concentrador consegue transportar, ou ainda um custo fixo associado a cada concentrador, além dos custos de alocação dos nós de demanda da rede aos concentradores.

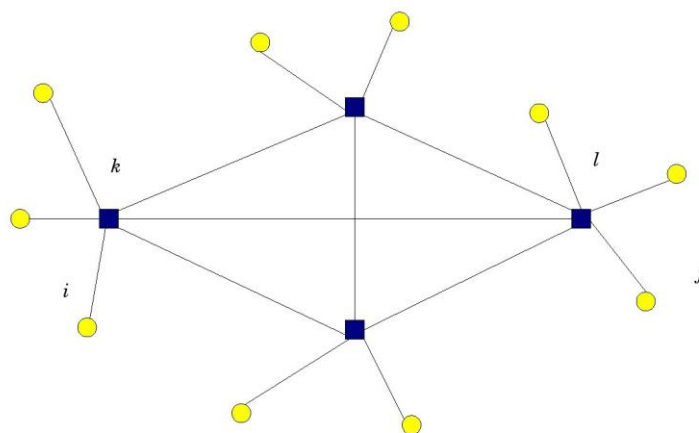


Figura 1 – Rede do tipo hub-and-spoke com n=14

Quando não existe restrição quanto ao fluxo (de pessoas ou de dados, por exemplo) que passa por um concentrador e cada nó de demanda não pode ser alocado a mais que um concentrador, o problema denomina-se Problema Não-Capacitado de Localização de Concentradores (USAHLP - *Uncapacitated Single Allocation Hub Location Problem*). Neste problema o número de concentradores é uma variável de decisão. No caso do número de concentradores ser fixo (por exemplo, igual a p), o problema é denominado de USApHMP (*Uncapacitated Single Allocation p -Hub Median Problem*) (CHEN, 2008; EBERY, 2001). No entanto, quando um nó de demanda pode ser alocado a mais do que um concentrador, o problema denomina-se problema de localização de concentradores com alocação múltipla (do inglês, *Uncapacitated Multiple Allocation Hub Location Problem*, UMAHLP) e quando existe restrição de capacidade quanto ao fluxo máximo de um concentrador, o problema é conhecido como problema de localização de concentradores capacitado (*Capacitated Single Allocation Hub Location Problem*, CSAHLP).

Neste trabalho pretende-se explorar o CSAHLP, e propor três soluções utilizando as meta-heurísticas: *Simulated Annealing*, VNS e Algoritmos Genéticos

2. Modelo Matemático

Goldman (1969) foi o primeiro a apresentar o problema de localização de concentradores. No entanto, o primeiro modelo matemático com função objetivo

quadrática para o problema de localização de concentradores é devido a O'Kelly (1987). A formulação matemática para o problema capacitado pode ser escrita como:

$$f(x) = \text{Min.} \sum_i \sum_j w_{ij} \sum_k \lambda d_{ik} x_{ik} + \sum_i \sum_j w_{ji} \sum_l \delta d_{jl} x_{jl} + \sum_i \sum_k x_{ik} \sum_j \sum_l \alpha x_{jl} d_{kl} w_{ij} + \sum_k f_k x_{kk} \quad (2.1)$$

Sujeito à:

$$\sum_k x_{ik} = 1, \quad \forall i \in V \quad (2.2)$$

$$x_{kk} - x_{ik} \geq 0, \quad \forall i, k \in V, \quad (2.3)$$

$$\sum_i E_i x_{ik} \leq Q_k x_{kk} \quad \forall k \in V \quad (2.4)$$

$$x_{ik} \in \{0,1\} \quad \forall i, k \in V. \quad (2.5)$$

Em que:

- V é conjunto de nós da rede;
- d_{ij} é a distância entre os nós i e j ;
- w_{ij} é a quantidade de fluxo transferido entre os nós i e j ;
- λ, α, δ são, respectivamente, os custos de coleta, transferência e distribuição;
- Q_i é a capacidade do concentrador i ;
- x_{ik} é uma variável de decisão, tal que $x_{ik} = 1$ se o nó i está alocado ao concentrador k ($x_{ik} = 0$ caso contrário). Deve-se observar que se $x_{kk} = 1$ então o nó k é um concentrador; Caso contrário, $x_{kk} = 0$.
- E_i é a soma dos fluxos que chegam em i . ($E_i = \sum_j w_{ij}$)

Nessa formulação, a função-objetivo (2.1) estabelece o custo total a ser minimizado que corresponde a soma dos custos de coleta transferência e distribuição de uma rede, a restrição, as restrições (2.2) garantem que cada nó de demanda será alocado a um único concentrador, as restrições (2.3) asseguram que as alocações serão feitas apenas para nós que são concentradores, as condições (2.4) restringem que o fluxo total que chega ao concentrador k não pode exceder a capacidade máxima de cada concentrador k e as restrições (2.5) correspondem às condições de integralidade das variáveis de decisão.

Ernst e Krishnamoorthy (1999) propõem dois algoritmos heurísticos para o problema capacitado: o primeiro baseia-se no método *simulated annealing* e o segundo em um método de descida randômico. Tal método consiste em gerar soluções vizinhas aleatoriamente e só aceitar movimentos de melhora. Com o apoio de um método do tipo *branch-and-bound* e com limitantes superiores providos pelas heurísticas os autores conseguiram obter soluções ótimas para problemas AP definidos em redes de até 50 nós,

com exceção do teste em que $n = 50$ e os custos fixos e a capacidade são do tipo apertados.

3. *Variable Neighbourhood Search*

O método de busca em vizinhança proposto por Mladenovic & Hansen (1997) é uma técnica de busca local que explora o espaço de soluções através de trocas sistemáticas de estruturas de vizinhança. Sua metodologia consiste em explorar vizinhanças gradativamente mais “distantes” da solução atual, a exploração de uma nova região somente acontece se um movimento de melhora é realizado. O pseudocódigo é apresentado pela Figura 2.

```
Procedimento VNS  
Seja  $s^0$  uma solução inicial e  $r$  o número de estruturas de vizinhança  
 $s \leftarrow s^0$  {Solução corrente}  
Enquanto (Critério de parada não satisfeito) faça  
   $k \leftarrow 1$ ; {Tipo de estrutura de vizinhança}  
  Enquanto ( $k \leq r$ ) faça  
    Gere um vizinho qualquer  $s' \in N^{(k)}(s)$   
     $s'' \leftarrow \text{BuscaLocal}(s')$   
    Se ( $f(s'') < f(s)$ )  
      então  $s \leftarrow s''$ ;  $k \leftarrow 1$   
      senão  $k \leftarrow k + 1$   
    Fim-se  
  Fim-enquanto  
Fim-enquanto  
Retorne  $s$   
Fim-procedimento
```

Figura 2 – Pseudocódigo VNS

A implementação deste método requer a definição de uma vizinhança e a escolha de um método de busca local. No entanto um vizinho deve ser definido de acordo com a representação da solução.

Para o problema considerado neste trabalho, a representação das soluções (ou, indivíduos do AG) baseiam-se na proposta de Topcuoglu *et al.* (2005) e corresponde a dois vetores de tamanho n : um para armazenar a localização dos *hubs* e outro para armazenar as alocações dos *spokes* aos *hubs*. Nestes vetores, denominados *HubArray* e *AssignArray*, cada posição corresponde a um nó da rede. O *HubArray* corresponde a um vetor binário em que cada posição armazena o valor 0, no caso do nó correspondente a esta posição ser um *spoke*, ou 1, no caso deste nó ser um *hub*. O *AssignArray* equivale a

um vetor em que cada posição armazena o índice do *hub*, ao qual o nó correspondente está associado.

As vizinhanças definida para este trabalho corresponde aos seguintes movimentos:

- *Swap Nodes*: reponsável pela escolha de dois nós não-hub para a troca de suas alocações;
- *Swap Hubs*: obtém dois concentradores e troca as associações de um concentrador com as associações de outro;
- *Reallocate Node*: escolhe um nó não hub e aloca a outro concentrador diferente de sua alocação original;
- *Reallocate Hubs*: pega um nó não hub e seu concentrador associado, e faz uma troca entre eles. Neste caso, o nó não hub passa a ser concentrador e o concentrador um nó não hub, com isso todas as associações ao antigo concentrador passa a se conectar ao concentrador atual.
- *New Hub*: escolhe um nó não hub e o transforma em um concentrador;
- *Delete Hub*: retira um concentrador da solução, e o transforma em um nó não hub. Os nós não hub alocados a este são realocados a outros concentradores de modo aleatório.

Em cada vizinhança definida o critério de escolha acontece de forma aleatória. Além disso, as soluções inviáveis em que o fluxo total ultrapassam a capacidade do concentrador são tratadas em todos os métodos aqui definidos através de penalização, por meio da atribuição de um custo elevado a solução. Com isso o método evita a escolha de soluções inviáveis.

O método de busca local utilizado para o VNS faz uma reordenação dos nós não hubs de maneira a realocá-los, na posição que gere o menor custo possível. A cada iteração do método é feito uma troca do nó não hub i com o nó não hub j , se o custo da solução diminuiu a troca de i com j é aceita, caso contrário a ordem original é mantida.

4. Simulated Annealing

O método *Simulated Annealing*, introduzido por Kirkpatrick *et al.* (1983), foi baseado no trabalho de Metropolis *et al.* (1953), nesta técnica a “temperatura” não é constante como no Metropolis. O processo consiste inicialmente em “fundir” o sistema a uma alta temperatura (este estado tem uma maior probabilidade de se aceitar soluções que pioram a função objetivo) e então, resfriar lentamente o sistema até que ele se “congele” e nenhuma mudança posterior possa ocorrer. Um pseudocódigo do método é apresentado pela Figura 3.

5. Algoritmos Genéticos

Os AGs são meta-heurísticas de busca inspiradas na teoria da evolução, capazes de encontrar boas soluções para um problema. Tal método, introduzido por Holland (1975), baseia-se em determinar em uma população de indivíduos (possíveis soluções

para o problema), aqueles que, por serem mais adaptados, irão se reproduzir e gerar descendentes para novas gerações.

```
Procedimento Simulated Annealing  
Entrada:  $T_0, T_f, L, \alpha$  (entre 0 e 1)  
 $T \leftarrow T_0 ; S_0$   
gera solução inicial;  
 $S \leftarrow S_0$ ;  
 $S^* \leftarrow S_0$   
enquanto  $T > T_f$  faça (temperatura alta)  
    para  $cont \leftarrow 1$  até  $L(T)$  faça (iterações para equilíbrio)  
         $S' \leftarrow$  seleciona uma solução vizinha de  $S$   
         $D_{custo} \leftarrow custo(S') - custo(S)$   
        se  $D_{custo} < 0$  ou  $U[0,1] < \exp(-D_{custo}/T)$   
            então  $S \leftarrow S'$   
            se  $(S < S^*)$  então  $S^* \leftarrow S$   
    fim do para  
     $T \leftarrow \alpha T$   
fim-enquanto  
fim-procedimento
```

Figura 3 – Pseudocódigo *Simulated Annealing*

Algoritmos Genéticos tradicionais podem ser descritos, resumidamente, pelos seguintes passos apresentados na Figura 4.

```
Passo 1. Crie uma população inicial aleatória de indivíduos (ou cromossomos) com tamanho  $N_p$ .  
Passo 2. Aplique a função de avaliação (fitness) a cada indivíduo, obtendo o valor da função objetivo para cada um.  
Enquanto (critério de parada não foi satisfeito) faça  
    Passo 3. Gere novos indivíduos a partir da população atual, através de operadores evolutivos como: crossover, mutação e elitismo.  
    Passo 4. Aplique a função de avaliação aos novos cromossomos.  
    Passo 5. Selecione os mais aptos, ou seja, os que mais melhoram a função objetivo, e substitua os indivíduos de menor aptidão na população.  
Fim-enquanto  
Passo 6. Retorne a solução mais apta
```

Figura 4 – Descrição do AG

Algoritmos Genéticos tradicionais podem ser descritos, resumidamente, pelos seguintes passos apresentados na Figura 4.

A população inicial é gerada aleatoriamente, sendo que o número de *hubs* deve respeitar um limitante superior, como no trabalho de Chen (2007). Para calcular esse limitante, gera-se uma solução viável qualquer com apenas 1 *hub* inicialmente, calcula-se os custos fixos e variáveis separadamente, e se a soma dos custos fixos f_j for maior que a soma dos custos variáveis (coleta, transferência e distribuição) tem-se o limitante igual a 1 *hub*. Caso contrário, aumenta-se o número de *hubs* de uma unidade, fazendo com que uma nova solução seja gerada e todo o procedimento se repita até que a soma dos custos fixos seja maior que a soma dos custos variáveis, ou o número de *hubs* atingir n (número de nós da rede).

O método de cruzamento (*crossover*) escolhido para este trabalho foi o de dois pontos, pelo motivo de sua natureza de funcionamento encorajar a exploração do espaço de soluções, tornando a convergência mais promissora (Silva, 2004).

O operador de *crossover* consiste no sorteio de dois pontos de cruzamento e na troca dos materiais genéticos, como mostrado na Figura 5. Os genes que estão entre os dois pontos de corte do primeiro cromossomo, unem-se com os genes que estão antes e depois do primeiro e segundo ponto de cruzamento, no outro cromossomo. Isto é realizado tanto para o *HubArray* quanto para o *AssignArray*. Após a troca dos genes, um operador se responsabiliza em verificar se o *spoke i*, no *AssignArray*, está associado a um *hub* válido do *HubArray*, caso contrário, este *spoke* deve ser realocado ao seu *hub* mais próximo.

O método de seleção utilizado foi a Seleção por *Ranking* Linear, que ordena os indivíduos de acordo com seu grau de adaptação (*fitness*), deixando o indivíduo mais adaptado na posição 1 e o menos adaptado na posição n . A probabilidade de seleção é designada a cada indivíduo linearmente, de acordo com sua posição no *ranking* (Silva, 2004).

O operador de mutação implementado neste trabalho é composto por três fases, denominadas *shift*, *exchange* e uma busca local (*Local Search*, LS). Os dois primeiros são uma versão estendida de parte do movimento de busca tabu usada por Abdnnour-Helm (1998), onde:

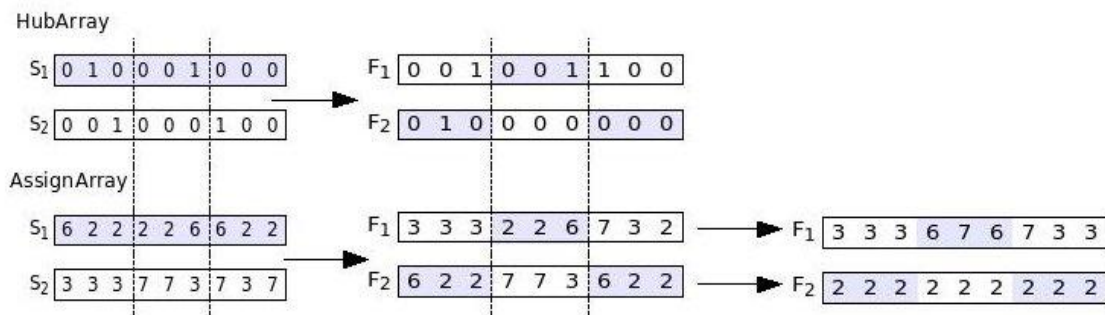


Figura 5 – Crossover de dois pontos

O *shift* sorteia um *spoke* e o associa a outro *hub*; se o cromossomo possuir apenas um *hub*, essa função não se aplica;

A função *exchange* seleciona dois *spokes* aleatoriamente e troca suas associações. Um pré-requisito para esse operador é a existência de pelo menos dois *hubs* e nós não *hubs*. No caso de haver apenas um *hub*, ou um único *spoke*, esse método não é executado;

6. Resultados

A verificação da eficiência dos métodos implementados, basearam-se no conjunto de dados AP (Ernest e Krishnamoorthy, 1996). Este conjunto tem sido referenciado em diversos trabalhos, como em Sasaki e Fukushima (2003) e Chen (2007).

O conjunto de dados AP (*Australian Post*), proposto por Ernest e Krishnamoorthy (1996), derivado do fluxo de e-mails na Austrália, compõe um conjunto de testes que vão de 10 a 50 nós para exemplares pequenos, e de 100 a 200 nós para exemplares grandes. Os valores dos custos de coleta, transferência e distribuição são respectivamente 3.0, 0.75 e 2. Os fluxos não são simétricos, ou seja, $w_{ij} \neq w_{ji}$, além disso, um cliente (*spoke*) pode enviar um e-mail a si próprio ($w_{ii} \neq 0$).

Para ambas as metaheurísticas implementadas, os testes foram executados em um computador com processador Core2-Duo 2.0 GHz, com 2 GB de memória RAM, sob o sistema operacional Linux.

Os parâmetros do AG foram definidos como: ($Np=100$), probabilidade de *crossover* ($Pc=0.80$), número de gerações ($Ng=200$), probabilidade de mutação ($Pm=0.10$), porcentagem de elite ($Pe=0.20$) e número de iterações da busca local ($NL=100$).

Os testes apresentados nesta seção consideram que os custos fixos e a capacidade dividem-se em duas categorias: frouxos (L) e apertados (T). Para o CSAHLP cada método foi executado 10 vezes. Nestas tabelas são utilizadas as seguintes legendas para as colunas:

- **Ótimo** representa a solução ótima conhecida para o problema;
- **Sol** representa a melhor solução encontrada pelo método implementado;
- **TE** representa o tempo médio de execução total do algoritmo, até que o critério de parada seja alcançado;
- **Desv** corresponde à variação das soluções encontradas em relação à média;
- **Gap** corresponde ao erro da melhor solução encontrada pela metaheurística em relação à melhor solução encontrada na literatura;

Os valores de **Desv** e **Gap** foram calculados com o objetivo de analisar a robustez e eficácia dos métodos implementados. Os valores de *Desv* e *Gap* são calculados pelas equações (6.1) e (6.2), em que $Sol_{média}$ corresponde à solução média, *Sol* corresponde à solução de menor custo encontrada pela meta-heurística nas dez execuções e *Melhor* equivale a solução ótima ou a solução de menor custo encontrada na literatura.

$$Desv = 100 \times \frac{(Sol_{media} - Sol)}{Sol} \quad (6.1)$$

$$Gap = 100 \times \frac{(Sol - Melhor)}{Melhor} \quad (6.2)$$

Tabela 1 – Resultados Iniciais

Teste	Ótimo	Algoritmo Genético				Simulated Annealing			
		Sol	Desv	Gap	TE	Sol	Desv	Gap	TE
AP10LL	224250,05	224250,05	0,00	0,00	0,47	225979,17	0,00	0,77	0,02
AP10LT	250992,26	250992,26	0,00	0,00	0,74	250992,26	3,27	0,00	0,02
AP10TL	263399,94	263399,95	0,00	0,00	0,49	277591,06	0,00	5,39	0,01
AP10TT	263399,94	263399,95	0,00	0,00	0,47	264543,96	0,00	0,43	0,02
AP20LL	234690,96	234690,97	0,00	0,00	0,83	234690,97	0,00	0,00	0,07
AP20LT	253517,40	253517,40	1,61	0,00	1,39	271091,71	0,00	6,93	0,07
AP20TL	271128,18	271128,18	0,00	0,00	0,77	278167,72	0,00	2,60	0,02
AP20TT	296035,40	296035,42	1,31	0,00	1,66	308665,25	5,73	4,27	0,08
AP25LL	238977,95	238977,96	0,00	0,00	0,72	248969,41	1,54	4,18	0,10
AP25LT	276372,50	276372,49	2,85	0,00	0,74	279990,93	7,95	1,31	0,11
AP25TL	310317,64	310317,65	0,00	0,00	0,45	310317,65	0,00	0,00	0,10
AP25TT	348369,15	348369,16	1,23	0,00	0,71	371948,26	3,70	6,77	0,15
AP40LL	241955,71	241955,70	0,01	0,00	0,23	241955,70	1,56	0,00	0,25
AP40LT	272218,32	272218,32	5,72	0,00	0,37	277727,35	7,32	2,02	0,39
AP40TL	298919,01	305552,00	0,15	2,22	0,20	299339,78	2,68	0,14	0,23
AP40TT	354874,10	362600,24	5,89	2,18	0,48	380707,80	6,54	7,28	0,39
AP50LL	238520,59	238520,58	0,00	0,00	0,24	238520,58	0,00	0,00	0,38
AP50LT	272897,49	275945,36	5,23	1,12	0,81	291916,81	3,25	6,97	0,70
AP50TL	319015,77	319268,98	1,42	0,08	0,33	325386,60	2,62	2,00	0,42
AP50TT	417440,99*	432389,49	10,70	3,58	0,73	426756,70	3,40	2,23	0,64

*A solução apresentada é a melhor solução encontrada até o momento.

Tabela 2 – Resultados Iniciais

Teste	Ótimo	VNS			
		Sol	Desv	Gap	Tempo
AP10LL	224250,05	224250,05	0,00	0,00	0,01
AP10LT	250992,26	260108,70	0,00	3,63	0,00
AP10TL	263399,94	264543,96	0,00	0,43	0,01
AP10TT	263399,94	263399,95	0,09	0,00	0,01
AP20LL	234690,96	234690,97	0,00	0,00	0,10
AP20LT	253517,40	255388,04	2,27	0,74	0,12
AP20TL	271128,18	271128,18	0,00	0,00	0,08
AP20TT	296035,40	318185,57	0,41	7,48	0,12
AP25LL	238977,95	245182,91	2,32	2,60	0,19
AP25LT	276372,50	279990,93	5,62	1,31	0,30
AP25TL	310317,64	310317,65	1,56	0,00	0,15
AP25TT	348369,15	352069,91	0,27	1,06	0,26
AP40LL	241955,71	242114,04	4,27	0,07	1,84
AP40LT	272218,32	277599,94	6,23	1,98	3,33
AP40TL	298919,01	298919,00	6,58	0,00	1,64
AP40TT	354874,10	363401,18	7,84	2,40	2,84
AP50LL	238520,59	238520,58	2,16	0,00	5,21
AP50LT	272897,49	272897,49	5,18	0,00	12,18
AP50TL	319015,77	323918,43	5,68	1,54	3,17
AP50TT	417440,99*	421336,42	11,21	0,93	7,79

Tabela 3 – Resumo dos resultados

	Média		
	Desv	Gap	Tempo
Algoritmo Genético	1,806	0,348	0,6415
Simulated Annealing	2,478	2,6645	0,2085
VNS	3,0845	1,2085	1,9675

Pelos testes apresentados nas Tabelas 1, 2 e 3 é possível verificar que o Algoritmo Genético foi mais eficiente e mais robusto em relação as outras abordagens propostas.

7. Considerações Finais

O problema de localização de concentradores Capacitado é um problema de Otimização Combinatória relevante, pois ocorre em diversas situações práticas em que o transporte de alguma entidade (pessoas, dados, produtos, etc) precisa passar por um processo de agregação antes de ser distribuída ao seu destino. Boas soluções para o problema podem representar ganhos econômicos significativos para muitos setores empresariais. A solução do problema consiste na determinação da quantidade necessária e na localização dos concentradores, além da alocação das demais instalações a estes concentradores, minimizando os custos envolvidos e gerenciando para que a instalação não exceda sua capacidade máxima suportada.

O problema, no entanto, é conhecido ser da classe NP-difícil e algoritmos exatos, para determinar a solução ótima do problema, precisam utilizar métodos enumerativos que exigem grande esforço computacional e podem ser impraticáveis para exemplares do problema de grandes dimensões. Os métodos propostos neste trabalho, por serem heurísticos, não garantem que a solução obtida seja ótima, mas, como observado pelos testes realizados, é capaz de gerar boas soluções em um tempo computacional razoável.

De acordo com as análises é possível dizer que Algoritmos Genéticos é capaz de explorar a região viável do problema de maneira eficiente por meio de seus operadores genéticos. No entanto sua implementação é muito complexa, dependendo da aplicação pode ser escolhido o *Simulated Annealing* que é mais simples de implementar e também tem capacidade de obter boas soluções.

Referências

- Abdinnour-helm, S. (1998) A hybrid heuristic for the uncapacitated hub location problem. *European Journal of Operational Research*, v. 106, p. 489-499.
- Aykin, T. (1994) Lagrangian relaxation based approaches to capacitated hub-and-spoke network design problem. *European Journal of Operational Research*, v. 79, n. 33, p. 501-523.
- Chen, J.F. (2007) A hybrid heuristic for the uncapacitated hub location problem. *Omega, The International Journal of Management Science*, v. 35, p. 211-220.
- Chen, J.F. (2008) A note on solution of the uncapacitated single allocation p-hub median problem. *Journal of the Chinese Institute of Industrial Engineers*, v. 25, n. 1, p. 11-17.
- Ebery, J. (2001) Solving a large Single allocation p-hub problems with two or three hubs. *European Journal of Operational Research*, v. 128, n. 2, p. 447-458.
- Ernst, A.T.; Krishnamoorthy, M. (1996) Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science*, v. 4, n. 3, p. 139-154.
- Ernst, A.T.; Krishnamoorthy, M. (1999) Solution algorithms for the capacitated single allocation hub location problem, *Annals of Or*, v. 86, p. 141-159.
- Goldman, A.J. (1969) Optimal location for centers in a network. *Transportation Science*, v. 3, p. 352-360.
- Holland, J. (1975) *Adaptation in natural and artificial systems*. Ann Arbor, Michigan: University of Michigan press.
- Kirkpatrick, S.; Gelatt Junior, C.D.; Vecchi, M. P. (1983) Optimization by Simulated Annealing. *Science*, New York, v.220, p.671-680.
- Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M.; Teller, A. H.; and Teller, E. (1953) Equation of state calculations by fast computing machines. *J. Chem. Phys.* **21**, 1087-1092.
- Mladenovic, N. E Hansen, P. (1997) Variable neighborhood Search. *Comps. & Opns. Res.* 24, 11, 1097-1100.

- O'Kelly, M. (1987) A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research*, v. 32, p. 393-404.
- Sasaki, M.; Fukushima, M. (2003) On the hub-and-spoke model with arc capacity constraints. *Journal of the Operations Research Society of Japan*, v. 46, n. 4, p. 409-428.
- Silva, M.R. (2004) *Uma contribuição ao problema de localização de terminais de consolidação no transporte de carga parcelada*. Dissertação de Mestrado. Escola Politécnica da Universidade de São Paulo.
- Topcuoglu, H.; Corut, F.; Ermis, M.; Ylmaz, G. (2005) Solving the uncapacitated hub location problem using genetic algorithms. *Computers and Operations Research*, v. 32, p. 967-984.