Aplicação do Método GRASP para Resolução do Problema de Localização de Máxima Cobertura Integrado ao Problema de Roteamento

Maria José Pinto Lamosa, Mônica Maria De Marchi e Carmen Lúcia Ruybal dos Santos
Instituto de Estudos Avançados (IEAv), Caixa Postal 6044 - CEP 12228-970 - São José dos Campos, SP.

Resumo — **Neste trabalho propomos tratar o problema de localização de máxima cobertura e o problema de roteamento de forma integrada. Em um trabalho anterior, formulamos o problema e, para verificar a viabilidade de utilizar a formulação proposta, realizamos alguns testes computacionais considerando exemplos gerados aleatoriamente. Os testes mostraram que, dependendo dos dados do problema a ser tratado, pode ser computacionalmente difícil utilizar o modelo para encontrar uma solução. Visando desenvolver métodos adequados (viáveis computacionalmente) para a resolução do problema integrado, o próximo passo consistiu em utilizar heurísticas, sendo a heurística GRASP o primeiro método utilizado.**

Palavras-chaves — **Problema de roteamento, MCLP, Formulação matemática, GRASP.**

I. INTRODUÇÃO

Problemas de localização, em geral, tratam de decisões sobre onde localizar facilidades em uma rede, considerando que existem clientes (demandas) a serem atendidos, de forma a otimizar um determinado critério [1,2]. O termo "facilidades" pode se referir à fábricas, depósitos, escolas, etc., enquanto o termo "clientes" pode se referir, respectivamente, a depósitos, unidades de vendas, estudantes, etc. Mais especificamente, no problema de localização de máxima cobertura (MCLP, do inglês Maximum Covering Location Problem), um número pré-definido de facilidades deve ser distribuído de forma a maximizar o número de demandas a serem atendidas.

O problema de roteamento consiste basicamente em estabelecer rotas eficientes para uma entidade com um determinado objetivo podendo, ou não, ser o ponto de partida igual ao ponto de chegada. Em geral, o objetivo é minimizar o custo total desta rota, o que inclui minimizar custos fixos, custos operacionais e, até mesmo, o número de entidades envolvidas. Este tipo de problema aparece em uma série de serviços como: entrega postal, entrega de mercadorias, rotas de ônibus ou de coleta de lixo, serviços emergenciais, entre outros [1,2].

Os problemas de localização e de roteamento são oriundos de importantes aplicações práticas e podem ser tratados de forma independente. Entretanto, existem situações e/ou aplicações onde tratá-los de forma integrada pode ser interessante.

M. J. P. Lamosa, maju@ieav.cta.br, Tel +55-12-39475342, M. M. De Marchi, monica@ieav.cta.br, Tel +55-12-39475343, C. L. R. dos Santos, carmenl@ieav.cta.br, Tel +55-12-39475339.

Neste trabalho, estamos interessados em tratar situações onde é possível fixar algumas das entidades (ou facilidades), permitindo que outras fiquem disponíveis para cobrir os eventuais "buracos" (demandas não atendidas) deixados pela cobertura gerada pelas entidades fixas. Nestes casos, as entidades fixas devem ser instaladas em posições que busquem minimizar a rota das entidades móveis, de forma a atender toda a demanda. A solução do problema indicará por qual das entidades cada demanda poderá ser atendida.

Para ilustrar esta situação, considere um sistema de vigilância e de monitoramento aéreo, onde o objetivo é cobrir (proteger) uma determinada área. Neste tipo de sistema é possível utilizar tanto as informações de radares fixos, quanto de radares móveis e aerotransportados. Para os radares móveis e fixos, estamos preocupados em posicioná-los de forma a gerar a melhor cobertura possível e, com relação aos aerotransportados, deseja-se gerar rotas eficientes para cobrir os corredores (buracos), resultantes do posicionamento dos radares fixos e/ou móveis. Para este caso, se tratarmos o problema de forma integrada é possível, dado o posicionamento dos radares fixos, posicionar os radares móveis de forma a minimizar a rota dos aerotransportados.

Para tratar o problema integrado considerando este enfoque, formulamos o problema matematicamente e utilizamos um pacote não comercial para resolvê-lo [6]. Os resultados dos testes computacionais realizados, considerando exemplos gerados aleatoriamente, mostraram a viabilidade de utilizar a formulação proposta. Entretanto, também confirmaram que, dependendo dos dados do problema a ser tratado, pode ser computacionalmente difícil utilizar o modelo para encontrar uma solução. Visando desenvolver métodos adequados (viáveis computacionalmente) para sua resolução, o próximo passo consistiu em utilizar métodos heurísticos, sendo nossa primeira sugestão utilizar o método GRASP (do inglês, *Greedy Randomized Adaptive Search Procedures*), tendo em vista o bom desempenho obtido, em trabalhos anteriores, ao aplicá-lo na resolução do MCLP [3]. Adaptações serão feitas para incorporar a parte do roteamento.

Na próxima seção apresentamos a formulação matemática sugerida para o problema integrado e, na seção 3, o algoritmo GRASP que está em fase de desenvolvimento. Na seção 4, apresentamos algumas considerações finais.

II. FORMULAÇÃO MATEMÁTICA

No modelo matemático global de programação inteira a ser apresentado, o objetivo consiste em atender toda a demanda utilizando facilidades fixas e móveis. No caso, as facilidades móveis serão utilizadas para cobrir os eventuais buracos deixados pelas facilidades fixas. Com isto, busca-se definir as melhores posições para localizar (fixar) um número pré-definido de facilidades, minimizando a rota das facilidades móveis.

Num primeiro momento, vamos considerar que temos uma única facilidade móvel e que a rota será definida sobre os pontos de demanda, ou seja, a ordem de atendimento de cada ponto de demanda definirá a rota que a facilidade móvel seguirá. Com isto, o problema pode ser formulado como a seguir.

minimizar: $$\sum_{i=1}^{M} \sum_{j=1}^{M} d_{ij} x_{ij} \qquad (1)$$

sujeito a: $$\sum_{j=1}^{M} x_{ij} - \sum_{j=1}^{M} x_{ji} = 0 \qquad i = 1,\ldots, M \qquad (2)$$

$$\sum_{j=1}^{N} y_j = p \qquad (3)$$

$$\sum_{j=1}^{M} x_{ij} - w_i = 0 \qquad i = 1,\ldots, M \qquad (4)$$

$$\sum_{j=1}^{N} a_{ij} y_j - z_i = 0 \qquad i = 1,\ldots, M \qquad (5)$$

$$w_i + z_i \geq 1 \qquad i = 1,\ldots, M \qquad (6)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1 \qquad \text{para os possíveis subconjuntos S} \qquad (7)$$

$$x_{ij}, w_i, z_i, y_k \in \{0,1\} \qquad i, j = 1,\ldots, M;\ k = 1,\ldots, N \qquad (8)$$

onde:

- M representa a quantidade de pontos de demanda a serem atendidos;
- N representa a quantidade de possíveis pontos de instalação de facilidades;
- p representa o máximo de facilidades a ser instalado;
- d_{ij} representa o custo de ir do ponto de demanda i para o ponto de demanda *j*;
- a_{ij} indica se o ponto de demanda i é coberto caso uma facilidade seja instalada no ponto *j*;
- x_{ij} variável de decisão que indicará se o arco (*i,j*) pertencerá ou não à rota gerada para a facilidade móvel;
- y_j variável de decisão que indicará se uma facilidade será instalada no ponto *j* ou não;
- w_i variável de decisão que indicará que o ponto de demanda *i* poderá ser atendido pela facilidade móvel;
- z_i variável de decisão que indicará que o ponto de demanda *i* poderá ser atendido pela facilidade fixa.

A função objetivo (1) definirá uma rota de custo mínimo. As restrições (2) representam o equilíbrio de fluxo nos nós de demanda, ou seja, se o nó i pertencer à rota, o fluxo deve entrar e deixar o nó *i*. A restrição (3) garante que exatamente *p* facilidades serão instaladas. A cobertura de toda a demanda é garantida pelas restrições (4), (5) e (6). Pois, se o arco (*i, j*) pertencer à rota gerada, as restrições (4) garantem que a demanda do nó *i* será atendida. Esta demanda também será atendida, pelas restrições (5), se for definido que a facilidade será instalada no ponto *j* e a demanda do ponto *i* estiver coberto por esta facilidade. As restrições (6) garantem que, pelo menos, uma das situações ocorrerá. O conjunto de restrições do tipo (7) é adicionado ao modelo para evitar a formação de ciclos. As restrições (8) exigem que as variáveis de decisão sejam binárias.

A aplicação desta formulação para resolução do problema de forma exata pode ser feita utilizando, por exemplo, pacotes comerciais como o pacote CPLEX [4] ou os não-comerciais como o GLPK [5, 6]. A dificuldade para utilizar este tipo de abordagem é que, à medida que o tamanho do problema aumenta (de acordo com os valores de *n*, *m* e *p*), a resolução do problema através de programação inteira torna-se computacionalmente difícil [6]. Objetivando encontrar métodos que resolvam o problema com um custo computacional viável optou-se por utilizar uma abordagem heurística, neste caso, a meta-heurística GRASP [2].

III. GRASP

A meta-heurística GRASP é um procedimento de busca aleatório míope e adaptativo e visa apresentar diferentes soluções através de um procedimento que consiste de duas fases: na primeira constrói-se uma solução inicial míope para o problema em análise a qual, na segunda fase, é submetida à busca local, visando melhorar a solução corrente.

Na literatura encontram-se vários mecanismos de construções míopes (ver [7, 8]), que consideram a inserção individual dos elementos na solução obtendo-se, a cada passo, uma solução parcial. No caso de uma função míope, por exemplo, cada candidato é escolhido pela sua contribuição para a solução parcial (também conhecida como função gulosa). Outra forma de escolha dos elementos pode ser feita aleatoriamente a partir de uma Lista Restrita de Candidatos (LRC), gerada de forma míope e dando ao processo uma característica probabilística. No caso da escolha aleatória, Resende e Velarde [7] apresentam alguns tipos de procedimentos que podem ser utilizados. A seguir, citamos alguns:

- LRC baseada na cardinalidade: o próximo candidato é escolhido de forma aleatória, a partir da lista míope gerada;
- LRC baseada em valor: a lista de candidatos é gerada a partir de uma função míope e de uma constante real **c**, cujo valor se encontra no intervalo 0 e 1 (se **c** = 0 o processo de seleção é míope; se **c** = 1 o processo de seleção é totalmente aleatório). Segundo Resende e Velarde [7], o valor de **c** dentro deste intervalo garante uma convergência rápida do algoritmo míope e uma diversidade de soluções.
- LRC aleatória e míope: neste processo, metade dos candidatos são escolhidos aleatoriamente e os demais através de um algoritmo míope;

Esta fase do processo permite que diferentes soluções sejam geradas a cada iteração GRASP. Mas estas soluções iniciais do GRASP não são necessariamente ótimos locais [7]. Como conseqüência, faz-se necessária a aplicação de um procedimento de busca local para tentar melhorar as soluções obtidas na fase construtiva. Esta busca realiza sucessivas trocas da solução corrente, sempre que uma melhor solução é encontrada na vizinhança. Este procedimento termina quando nenhuma solução melhor é encontrada. O critério de parada pode ser o número máximo de iterações ou o tempo máximo de execução.

A aplicação do GRASP em diferentes tipos de problemas tem crescido nos últimos anos. Festa e Resende [8] desenvolveram e disponibilizaram uma *webpage* com uma lista de trabalhos, *softwares* e aplicações utilizando o método.

Para resolver o problema integrado através da meta-heurística GRASP as seguintes variáveis, além das já consideradas, precisam ser definidas.

LRC	Lista de p posições, dentre as N possíveis posições de instalação das facilidades;
S	Vetor de p posições que conterá parte da solução do problema referente ao posicionamento das facilidades fixas;
S^*	Vetor que conterá as p posições do vetor S e a seqüência de pontos de demanda a fazer parte da rota;
C	Solução do problema integrado, ou seja, consistirá do custo da rota gerada em S^*;
Tabu	Lista com p posições que conterá os pontos de instalação de radares que são tabu, ou seja, que não poderão fazer parte da solução.
it	Contador para o número de iterações;
max_it	Número máximo de iterações do método;
max_it_igual	Número máximo de iterações que a solução corrente não sofre alteração.

O mecanismo de construção míope da LRC, utilizado neste trabalho, gera a cada iteração um valor aleatório que define o processo de seleção tentando garantir uma diversidade na LRC. Para gerar melhores soluções, na segunda fase do GRASP, utilizaremos uma lista Tabu. A seguir é apresentado o pseudocódigo do algoritmo GRASP para resolução do problema integrado considerando o enfoque descrito anteriormente.

Faça: $S = \{\ \}$, $S^* = \{\ \}$, $LRC = \{\ \}$, $Tabu = \{\ \}$, $it = 0$

1. Ordene os N possíveis pontos de instalação de facilidades, de forma crescente, de acordo com a quantidade de pontos de demanda que cada localização cobre;
2. Escolha, aleatoriamente, p posições dentre as N posições disponíveis de instalação de facilidades e coloque-as na lista *LRC*;

{Construção da Solução Inicial}

3. Para cada posição $r \in LRC$, calcule a cobertura gerada pela união dos elementos do conjunto S com a posição r. Considere *r_max* a posição que retorna a maior cobertura;
4. Retire a posição *r_max* da lista *LRC* e coloque em *S*;
5. Determine uma nova posição para ser colocada na lista *LRC* no lugar de *r_max*. Para isto, verifique qual dentre as posições ordenadas poderia ser considerada. A posição somente não pode ser considerada se for uma posição *Tabu* ou se já estiver nas listas *LRC* ou *S*;
6. Se o vetor S estiver com p posições, inclua este vetor em S^*. Senão, retorne ao passo 3;
7. Gere uma rota entre os pontos de demanda que não são cobertos pelas p posições do vetor S. Inclua esta rota no vetor solução S^*;
8. Defina o valor de C, calculando o custo da rota. Considere a rota definida em S^* como sendo a melhor até o momento.

{Busca Tabu}

9. Coloque a k-ésima posição na lista *Tabu*, onde k é a posição de S que, ao ser eliminada de S, resulta ainda na maior área coberta;
10. Insira a r-ésima posição da lista *LRC* no vetor S, onde r é a posição da lista *LRC* que, se for colocada em S retorna a maior área coberta;
11. Determine uma nova posição para ser colocada na lista *LRC* no lugar de r. Para isto, verifique qual dentre as posições ordenadas poderia ser considerada. A posição somente não pode ser considerada se for uma posição *Tabu* ou se já estiver nas listas *LRC* ou *S*;
12. Gere uma rota entre os pontos de demanda que não são cobertos pelas p posições do vetor S e calcule o custo desta rota;

{Atualização da Solução}

13. Atualiza a solução final (vetor S^* e variável C), se a rota gerada no passo 12 possuir um custo menor do que a solução armazenada. Neste caso, o vetor S^* consistirá do vetor S atual e da última rota gerada;
14. Faça $it = it + 1$. Se $it = max\_it$, pare o algoritmo e apresente a melhor solução obtida até o momento;
15. Se a solução gerada não foi atualizada por *max_it_igual* retorne ao passo 2, senão retorne ao passo 9.

O algoritmo ainda está em fase de desenvolvimento sendo que a implementação já foi iniciada.

IV. Considerações Finais

Neste trabalho, propomos um algoritmo heurístico, baseado na metodologia do GRASP, para resolver de forma integrada, o problema de localização de máxima cobertura e o problema de roteamento. Nossa expectativa é que o método proposto seja capaz de encontrar boas soluções para o problema integrado num tempo computacional aceitável, tendo em vista que testes computacionais realizados anteriormente [6] utilizando a formulação apresentada na seção II mostraram que sua aplicação no problema é viável mas, dependendo dos dados do problema considerado, encontrar a solução pode ser computacionalmente difícil.

Referências

[1] M. Daskin, *Network and Discrete Location: Models, Algorithms and Applications*, Wiley Interscience, New York, EUA, 1995.

[2] M. C. Goldbarg e H. P. L. Luna, *Otimização Combinatória e Programação Linear: Modelos e Algoritmos*, Ed. Campus, R.J., 2000.

[3] M. M. De Marchi, M. J. P. Lamosa, F. L. L. Medeiros e Santos, C. L. R. “Aplicação do Método GRASP no Problema de Posicionamento de Radares de Vigilância”, In: XXXVII Simpósio Brasileiro de Pesquisa Operacional (SBPO), Gramado, R.S., pp. 1295-1304, 2005.

[4] Ilog Cplex Reference Manual. Disponível em: < http://www.rpi.edu/dept/math/mathprogramming/cplex66/sun4x_56/doc/refman/onlinedoc/>. Acesso: 18/09/2008.

[5] GLPK 4.28. GNU Linear Programming Kit. Disponível em: <http://www.gnu.org/software/glpk/glpk.html>. Acesso: 02/04/2008.

[6] M. J. P. Lamosa, M. M. De Marchi e Santos, C. L. R. “O Problema de Localização de Máxima Cobertura Integrado ao Problema de Roteamento”, In: XL Simpósio Brasileiro de Pesquisa Operacional (SBPO), João Pessoa, P.B., 2005.

[7] M. G. C. Resende e J. L. G. Velard, “GRASP: Greedy Randomized Adaptive Search Procedures”, *Revista Iberoamericana de Inteligencia Artificial*, vol. 19, pp. :61-76, 2003.

[8] P. Festa e M. G. C. Resende, “GRASP: an Annotated of Bibliography”, Disponível em: <http://www.graspheuristic.org/>. Acesso: 12/06/2005.