UM MÉTODO HÍBRIDO PARA O PROBLEMA PROBABILÍSTICO DE LOCALIZAÇÃO-ALOCAÇÃO DE MÁXIMA COBERTURA

Marcos Antonio Pereira
Universidade Estadual Paulista
Av. Dr. Ariberto Pereira da Cunha, 333 - 12516-410 - Guaratinguetá - SP
mapereira@feg.unesp.br

Leandro Callegari Coelho
Université Laval
2325 de la Terrasse - G1K 0A6 - Québec City, Canada
leandro.coelho@cirrelt.ca

Luiz Antônio Nogueira Lorena
Ligia Corrêa de Souza
Instituto Nacional de Pesquisas Espaciais
Av. dos Astronautas, 1758 - 12221-010 - São José dos Campos - SP
lorena@lac.inpe.br, li.correasouza@gmail.com

RESUMO

Este artigo apresenta um algoritmo híbrido para resolver o Problema Probabilístico de Localização-Alocação de Máxima Cobertura. Explorando a formulação matemática do problema, uma heurística flexível de busca em vizinhança é empregada para obter as soluções de localização, enquanto um método exato fornece a respectiva solução ótima dos subproblemas de alocação. Testes computacionais realizados com dados da literatura permitiram avaliar a eficiência do método proposto. O problema inteiro foi modelado de forma eficiente para ser resolvido de modo exato, o que possibilitou encontrar novas melhores soluções para 19 das instâncias testadas, comprovando a otimalidade para 18 delas. O método híbrido apresentou desempenho consistente, encontrando 94,5% das melhores soluções conhecidas na literatura.

PALAVRAS CHAVE. Localização de facilidades, sistemas congestionados, matheurística.

Área Principal: MH – Metaheurísticas, OC – Otimização Combinatória, PM – Programação Matemática.

ABSTRACT

This paper presents a hybrid algorithm to solve the Probabilistic Maximal Covering Location-Allocation Problem. A linear programming formulation presents location variables, that are determined by a flexible neighborhood search heuristic, and allocation variables, obtained as optimal solutions of an easier subproblem. Computational experiments on a set of benchmark instances from the literature were conducted to evaluate the efficiency of the proposed method. The integer problem formulation was efficiently implemented to be solved exactly, obtaining 19 new best solutions and proving optimality for 18 of them. The hybrid method performed consistently, finding the best known solutions for 94.5% of the instances.

KEYWORDS. Facility location, congested systems, matheuristic.

Main Area: MH – Metaheuristics, OC – Combinatorial Optimization, PM – Mathematical Programming.

1. Introdução

A localização de facilidades desempenha um papel importante na área de Logística. O número de empresas que buscam métodos quantitativos para estimar a melhor forma de atender a demanda por produtos ou serviços cresce a cada dia. Em alguns casos, a disponibilidade de atendimento está associada à distância ou ao tempo de deslocamento dos clientes até uma facilidade existente. Nestes casos, o decisor deve determinar a melhor configuração para a instalação de facilidades de forma a atender a maior parte da demanda.

O Problema de Localização de Máxima Cobertura (MCLP, na sigla em inglês) foi proposto por Church e ReVelle (1974) e é um problema clássico de localização de facilidades que visa selecionar um determinado número de locais candidatos para a instalação de facilidades de forma a maximizar a demanda total de clientes que estejam dentro do raio de atendimento de uma facilidade instalada. No setor público, este modelo é utilizado, por exemplo, para determinar a localização de serviços de emergência, como estações de ambulância e brigadas de combate a incêndio. No setor privado, busca-se determinar, por exemplo, a localização de caixas bancários de auto-atendimento, postos de gasolina, lojas de redes de *fast-food*, antenas de telefonia celular, etc. Uma lista mais abrangente de aplicações pode ser encontrada em Hale e Moberg (2003), Galvão (2004) e Serra e Marianov (2004). Variações deste problema podem ser encontradas em Pirkul e Schilling (1991) e Berman *et al.* (2009; 2013). As propostas de solução para o MCLP incluem heurísticas gulosas (Church e ReVelle, 1974; Daskin, 1995), relaxações de programação linear (Church e ReVelle, 1974) e lagrangeana (Galvão e ReVelle, 1996), algoritmos genéticos (Arakaki, 2001), heurística lagrangeana/*surrogate* (Lorena e Pereira, 2009) e geração de colunas (Pereira *et al.*, 2007).

Em algumas aplicações, o número de clientes (e sua respectiva demanda) alocados a uma facilidade pode causar algum impacto no comportamento do sistema: dependendo da natureza do serviço, os clientes podem ter que esperar para serem atendidos. Em tais sistemas congestionados, a qualidade do serviço é medida não apenas pela proximidade a uma facilidade, mas também por algum critério de desempenho, como o número de clientes na fila ou o tempo de espera para atendimento. Assumindo que o tempo médio de atendimento seja constante no tempo, pode-se incluir restrições de capacidade das facilidades à formulação matemática do MCLP, mas isso pode incorrer em facilidades ociosas ou sobrecarregadas.

Para tratar de cenários mais realistas, Marianov e Serra (1998) propuseram um modelo que considera que a chegada dos clientes a uma facilidade pode ser modelada como um processo estocástico, com taxa baseada no valor da demanda de cada cliente. Estimando-se o número esperado de clientes na fila ou o tempo máximo de espera para atendimento, os autores definem um limite mínimo para a qualidade do serviço, resultando num modelo que é uma extensão do MCLP, conhecido como Problema Probabilístico de Localização-Alocação de Máxima Cobertura (PMCLAP ou QM-CLAM, na sigla em inglês). Devido à complexidade deste problema e ao tamanho das instâncias reais, a maioria dos métodos de resolução são baseados em heurísticas. Corrêa *et al.* (2008) propõe o emprego da heurística *Clustering Search* para detectar regiões promissoras de busca baseadas em agrupamentos de soluções e Corrêa *et al.* (2009) apresentam um método baseado em grafos de conflito e geração de colunas.

Os modelos de problemas de localização e alocação contém, pelo menos, dois tipos de variáveis de decisão: onde instalar uma facilidade (decisão de localização) e quais clientes atender pela facilidade instalada (decisão de alocação). Os métodos tradicionais de solução, como *branch-and-bound*, determinam as decisões de localização e alocação simultaneamente, alocando clientes a locais candidatos que ainda estão sendo avaliados para a instalação de uma facilidade. Entretanto, uma abordagem hierarquizada parece ser um processo mais natural num método heurístico: primeiramente determina-se a localização das facilidades e só então faz-se a alocação dos clientes às mesmas.

Este artigo apresenta um novo método iterativo híbrido que explora esta característica do problema. As decisões de localização serão tratadas pela metaheurística de busca em vizinhança *adaptive large neighborhood search* (ALNS) e as decisões de alocação correspondentes serão obtidas como soluções ótimas de um subproblema inteiro. A metaheurística ALNS foi proposta por Ropke e Pisinger (2006a) para a resolução de uma classe de problemas de roteamento de veículos (Ropke e Pisinger, 2006b) e, desde então, tem sido adaptada para tratar outros tipos de problemas, como sequenciamento de veículos (Bartodziej *et al.*, 2009; Pepin *et al.*, 2009), roteamento de veículos com estoques (Coelho *et al.*, 2012a; 2012b) e tabela de horário de trens (Barrena *et al.*, 2013). Até esta data, sabe-se de apenas uma aplicação do ALNS a um problema com um componente de localização (Hemmelmayr *et al.*, 2012), que é significantemente diferente do problema abordado neste artigo.

No restante deste artigo será apresentada a pesquisa desenvolvida. A Seção 2 apresenta a formalização e a formulação matemática do PMCLAP. A metaheurística híbrida ALNS é descrita na Seção 3. Os resultados dos testes computacionais são apresentados na Seção 4 e as conclusões compõem a Seção 5.

2. Descrição do Problema e Formulação Matemática

O MCLP visa localizar p facilidades dentre n possíveis locais candidatos, de forma a maximizar a demanda atendida para um dado raio de cobertura S. Entretanto, este problema não considera a capacidade de atendimento das facilidades ou o congestionamento do sistema. Para tratar dessa limitação, Marianov e Serra (1998) introduziram o PMCLAP, uma extensão do MCLP que impõe um limite mínimo na qualidade do atendimento. A medida de qualidade do serviço de uma facilidade é calculada em função do número de clientes na fila ou do tempo de espera para atendimento.

Formalmente, o problema é definido como um grafo completo com um conjunto $\mathcal{N}$ de n vértices. A cada vértice $i \in \mathcal{N}$ está associada uma demanda d_i e uma distância de serviço ou raio de atendimento S_i, caso uma facilidade seja instada no vértice i. Sem perda de generalidade, pode-se considerar um raio de atendimento S para todas as facilidades. Seja $\mathcal{N}_i$ o subconjunto dos vértices a até S unidades de distância do vértice i, ou seja, o conjunto de locais candidatos j que podem atender o cliente i. Para o PMCLAP, o parâmetro f_i representa o impacto do cliente i no congestionamento do sistema, sendo calculado como uma fração de d_i. Assume-se que os clientes chegam às facilidades de acordo com uma distribuição de Poisson com taxa μ e o parâmetro α é a probabilidade mínima de um cliente, ao chegar à facilidade designada, encontrar uma fila com b clientes ou ter que aguardar τ minutos para ser atendido.

Para modelar o PMCLAP, define-se as variáveis de decisão binárias y_j, que assumem o valor 1 se, e somente se, uma facilidade for instalada no local candidato $j \in \mathcal{N}$ e x_{ij}, que assumem o valor 1 se, e somente se, a demanda do cliente $i \in \mathcal{N}$ for atendida pela facilidade instalada em j. Dessa forma, o problema pode ser formulado como:

$$\text{maximizar} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}_i} d_i x_{ij} \tag{1}$$

sujeito a

$$\sum_{j \in \mathcal{N}_i} x_{ij} \leq 1 \quad i \in \mathcal{N} \tag{2}$$

$$\sum_{j \in \mathcal{N}} y_j = p \tag{3}$$

$$x_{ij} \leq y_j \quad i \in \mathcal{N} \quad j \in \mathcal{N} \tag{4}$$

$$\sum_{i \in \mathcal{N}} f_i x_{ij} \leq \mu \sqrt[b+2]{1-\alpha} \quad j \in \mathcal{N} \tag{5}$$

ou

$$\sum_{i \in \mathcal{N}} f_i x_{ij} \leq \mu + \frac{1}{\tau} \ln(1-\alpha) \quad j \in \mathcal{N} \tag{6}$$

$$y_j, x_{ij} \in \{0, 1\} \quad i \in \mathcal{N} \quad j \in \mathcal{N}. \tag{7}$$

A função objetivo (1) maximiza a demanda total atendida. As restrições (2) garantem que cada cliente seja atendido por apenas uma facilidade. A restrição (3) define o número de facilidades a serem instaladas. As restrições (4) impõem que os clientes sejam atendidos apenas pelas facilidades instaladas. A natureza probabilística do problema é representada pelas restrições (5) e (6): as restrições (5) garantem que a facilidade j tenha, no máximo b clientes na fila, com probabilidade mínima α e as restrições (6) garantem que o tempo de atendimento na facilidade j é de, no máximo, τ minutos, com probabilidade mínima α. As restrições (7) definem a natureza binária das variáveis. É óbvio que $x_{ij} = 0$ para todo $j \notin \mathcal{N}_i$. O PMCLAP é *NP-hard* (Pirkull e Schilling, 1991), o que implica que este modelo pode ser resolvido de forma exata e com mínimo esforço computacional apenas para instâncias relativamente pequenas, como será apresentado na Seção 4.

Observe que, sem as restrições (5) e (6), o modelo resultante corresponde ao MCLP.

3. Algoritmo Híbrido de Busca em Vizinhança

Considerando separadamente as decisões de localização e de alocação, foi desenvolvido um método iterativo híbrido para maximizar o atendimento da demanda. A cada iteração, os valores das variáveis de localização são obtidas por uma heurística flexível, baseada na metaheurística ALNS, que contém um subproblema inteiro para determinar os valores ótimos das variáveis de alocação e o valor da função objetivo.

A Figura 1 mostra a representação de uma solução de localização para uma rede com $n = 10$ locais candidatos e $p = 3$ facilidades instaladas.

y:	0	1	0	0	1	0	0	0	1	0
	1	2	3	4	5	6	7	8	9	10

Figura 1: Representação da variável de localização y

Em termos gerais, o ALNS destrói e repara partes de uma solução, num processo iterativo, buscando encontrar soluções melhores. Para isso, aplicam-se operadores de destruição e de reparo, que competem entre si: operadores que produziram soluções melhores no passado têm mais chance de serem utilizados. Utilizando conhecimento do problema, foram definidos operadores de destruição e reparo que atuam fechando e depois abrindo facilidades. Com as decisões de localização fixadas, define-se um subproblema inteiro para calcular o valor ótimo das decisões de alocação, a cada iteração. Tal algoritmo pode ser caracterizado como uma *matheurística* (Maniezzo *et al.*, 2009), ou seja, uma hibridização entre um algoritmo heurístico e um método exato de programação matemática.

A escolha dos operadores é baseada numa roleta, com probabilidades que dependem do desempenho anterior dos mesmos. Tal medida de desempenho também é utilizada para definir o *score* π_r e o peso ω_r para cada operador r. Assim, para h operadores, o operador $\bar{r}$ será selecionado com probabilidade $\omega_{\bar{r}} / \sum_{r=1}^{h} \omega_r$. Inicialmente, todos os pesos são fixados em 1 e todos os *scores* são fixados em 0. A cada iteração, o *score* do operador selecionado é atualizado, sendo incrementado de σ_1 se o operador produz uma nova melhor solução, de σ_2 se produzir uma solução melhor que a solução incumbente e de σ_3 se a solução não é de melhoria mas ainda é aceita como nova solução corrente. A cada θ iterações (segmento), os pesos são atualizados de acordo com os *scores* obtidos no último segmento. Seja o_{rs} o número de vezes que o operador r foi utilizado no último segmento s. Os pesos atualizados são calculados como:

$$\omega_r = \begin{cases} \omega_r & \text{se } o_{rs} = 0 \\ (1-\eta)\omega_r + \eta\pi_r / o_{rs} & \text{se } o_{rs} \neq 0, \end{cases} \tag{8}$$

onde $\eta \in [0, 1]$ é denominado fator de reação e controla a rapidez com que os pesos são ajustados de acordo com as mudanças de desempenho do operador. Ao final de cada segmento, os *scores* dos operadores são reinicializados em 0.

Como ocorre em outras implementações do ALNS (Coelho *et al.*, 2012b, Ropke *et al.*, 2006a; 2006b), utiliza-se um critério de aceitação de soluções baseado na heurística Simulated Annealing. Seja $z(\cdot)$ o valor da função objetivo da solução $\cdot$. Dada uma solução s, uma solução vizinha s' será sempre aceita se $z(s') > z(s)$; caso contrário, será aceita com probabilidade $e^{(z(s)-z(s'))/T}$, onde $T > 0$ é a temperatura atual. A temperatura tem valor inicial $T_{inicial}$, decrescendo a cada iteração de acordo com um fator de resfriamento ϕ, onde $0 < \phi < 1$.

As principais características do algoritmo serão descritas nas subseções seguintes.

3.1. Solução inicial

O algoritmo pode iniciar com uma solução vazia ou arbitrária. Se a solução inicial for vazia então utiliza-se apenas operadores de reparo na primeira iteração visando abrir p facilidades. Na solução inicial arbitrária seleciona-se ao acaso p locais candidatos para a instalação das facilidades. Vários trabalhos demonstraram que a qualidade da solução inicial não influencia o desempenho global da metaheurística ALNS (Coelho *et al.*, 2012a; 2012b; Barrena *et al.*, 2013).

3.2. Lista de operadores

As características do problema em questão foram cuidadosamente consideradas para definir os operadores. Dada uma solução de localização, os operadores de destruição e de reparo irão fechar ou abrir facilidades, respectivamente, visando explorar diferentes vizinhanças.

3.2.1. Operadores de destruição

1. Fechar arbitrariamente ρ facilidades

Este operador fecha ρ facilidades, ao acaso. Aqui, ρ é um número aleatório gerado de acordo com uma distribuição semi-triangular com inclinação negativa, definida em $[1, p]$. Esta distribuição é útil para o refinamento de soluções, visto que causa poucas alterações na mesma quando ρ é pequeno, o que ocorre com mais freqüência devido às características desta distribuição. Entretanto, quando ρ é grande, as mudanças na solução são maiores.

2. Fechar a facilidade com o menor número de clientes

Este operador identifica, dentre as facilidades abertas, aquela que possui o menor número de clientes alocados e a fecha. Neste caso, considera-se o número de clientes em potencial que estão dentro do raio de atendimento S de cada facilidade aberta j. Este operador é útil para fechar facilidades com pouco potencial de atendimento, permitindo que outra facilidade com maior potencial de atendimento seja aberta.

3. Fechar a facilidade com a menor demanda atendida

Este operador fecha a facilidade com o menor potencial de atendimento da demanda. É similar ao operador anterior, mas neste caso considera-se a demanda dos clientes dentro do raio de atendimento S de cada facilidade aberta j.

4. Fechar uma das duas facilidades mais próximas

Este operador identifica as duas facilidades mais próximas na solução atual e fecha uma delas, ao acaso. Este operador busca espalhar as facilidades, permitindo assim que mais clientes sejam atendidos.

3.2.2. Operadores de reparo

Todos os operadores de reparo calculam o número de facilidades a serem abertas, sendo executados repetidamente até que a solução produzida contenha p facilidades.

1. Instalar aleatoriamente uma facilidade

Este operador seleciona, ao acaso, um local candidato para instalar uma facilidade. É particularmente útil para diversificar a busca por melhores soluções.

2. Instalar uma facilidade a, no mínimo, $2S$ unidades das facilidades já abertas

Este operador busca um local candidato que esteja a mais de $2S$ unidades de distância das facilidades já instaladas, para a instalação de uma nova facilidade. Caso não haja tal local candidato, seleciona-se o local candidato mais distante das facilidaes já instaladas. O propósito deste operador é buscar atender os clientes não atendidos pelas facilidades existentes.

3. Instalar a facilidade com o maior potencial de atendimento (clientes)

Este operador avalia, dentre os locais candidatos não utilizados, aquele que atende o maior número de clientes ainda não atendidos pelas facilidades existentes. Com isso, busca-se atender clientes não atendidos pelas facilidades existentes, permitindo a sobreposição das áreas de atendimento das facilidades, o que pode melhorar a qualidade do serviço, sendo portanto útil para a parte probabilística do problema. Na hipótese de que todos os clientes já estejam atendidos, seleciona-se um local candidato, ao acaso.

4. Instalar a facilidade com o maior potencial de atendimento (demanda)

Semelhante ao operador anterior, mas buscando pelo local candidato capaz de atender a maior demanda total ainda não atendida pelas facilidades existentes. Na hipótese de que todos os clientes já estejam atendidos, seleciona-se um local candidato, ao acaso.

3.3. Solução do subproblema

Uma vez que a heurística ALNS terminou de destruir e reparar a solução de localização deve-se determinar a solução de alocação considerando as restrições probabilísticas e calcular o valor da função objetivo. Para isto, deve-se resolver de forma exata o seguinte problema inteiro no qual as variáveis y_j são substituídas pelos valores $\bar{y}_j$ obtidos pela heurística ALNS:

$$\text{maximizar} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} d_i x_{ij} \tag{9}$$

sujeito a

$$\sum_{j\in\mathcal{N}} x_{ij} \leq 1 \quad i \in \mathcal{N} \tag{10}$$

$$x_{ij} \leq \bar{y}_j \quad i \in \mathcal{N}_j \quad j \in \mathcal{N} \tag{11}$$

$$\sum_{i\in\mathcal{N}_j} f_i x_{ij} \leq \mu \sqrt[b+2]{1-\alpha} \quad j \in \mathcal{N} \tag{12}$$

ou

$$\sum_{i\in\mathcal{N}_j} f_i x_{ij} \leq \mu_j + \frac{1}{\tau} \ln(1-\alpha) \quad j \in \mathcal{N} \tag{13}$$

$$x_{ij} \in \{0,1\} \quad i \in \mathcal{N} \quad j \in \mathcal{N}. \tag{14}$$

Note que como $\bar{y}$ é uma solução de localização viável, a restrição sobre o número de facilidades a serem instaladas pode ser removida da formulação, sendo considerada implicitamente. Este problema é significativamente mais fácil de ser resolvido do que o problema (1)−(7). A cada iteração, é necessário apenas atualizar os limites das variáveis x_{ij} nas restrições (11), podendo-se utilizar de técnicas de reotimização uma vez que apenas uma pequena parte do problema é alterada a cada iteração.

3.4. Procedimento de melhoria

Foi desenvolvido um procedimento para melhoria de soluções obtidas durante o processo que é aplicado sempre que o ALNS encontra uma nova melhor solução e a cada θ iterações. Neste procedimento, define-se um modelo de programação inteira utilizando a melhor solução de alocação conhecida, permitindo o fechamento de duas facilidades e a seleção de outros dois locais candidatos para a instalação de novas facilidades. Este procedimento pode ser considerado como uma busca em vizinhança na qual todas as combinações de fechamento de duas facilidades e abertura de duas novas facilidades são exploradas. Se o modelo produzir uma nova melhor solução, ela será utilizada para atualizar o ALNS. Neste modelo, $\bar{y}$ é um vetor binário que indica se a facilidade j está aberta (1) ou fechada (0) na melhor solução conhecida até o momento.

$$\text{maximizar} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} d_i x_{ij} \tag{15}$$

sujeito a

$$\sum_{j \in \mathcal{N}} y_j = p \tag{16}$$

$$\sum_{j \in \mathcal{N}} x_{ij} \leq 1 \quad i \in \mathcal{N} \tag{17}$$

$$x_{ij} \leq y_j \quad i \in \mathcal{N}_j \quad j \in \mathcal{N} \tag{18}$$

$$\sum_{i \in \mathcal{N}_j} f_i x_{ij} \leq \mu \sqrt[b+2]{1-\alpha} \quad j \in \mathcal{N} \tag{19}$$

ou

$$\sum_{i \in \mathcal{N}_j} f_i x_{ij} \leq \mu_j + \frac{1}{\tau} \ln(1-\alpha) \quad j \in \mathcal{N} \tag{20}$$

$$\sum_{j \in \mathcal{N}} \bar{y}_j y_j = p - 2 \tag{21}$$

$$y_j, x_{ij} \in \{0,1\} \quad i \in \mathcal{N} \quad j \in \mathcal{N}. \tag{22}$$

Neste problema, (15)−(20) são iguais à (1)−(6). A restrição adicional (21) garante que $p-2$ das facilidades instaladas permaneçam abertas. O modelo deve abrir duas facilidades fechadas de forma a atender à restrição (16). Este procedimento de busca local não foi implementado como um operador do ALNS por ser significativamente mais complexo que os demais operadores.

3.5. Definição de parâmetros e pseudocódigo

Os valores dos parâmetros utilizados nesta implementação do ALNS foram definidos após uma fase inicial de testes. O número máximo de iterações i_{max} depende da temperatura inicial $T_{inicial}$ e da taxa de resfriamento ϕ. Estes parâmetros foram definidos como:

$$T_{inicial} = 30000 \tag{23}$$

$$\phi = (0,01/T_{inicial})^{1/i_{max}} \tag{24}$$

Dessa forma, a taxa de resfriamento é definida em função do número desejado de iterações, ajustando de forma automática a probabilidade de aceitação de soluções piores pelo ALNS. O critério de parada é atingido quando a temperatura atingir 0,01. Nesta implementação, o número máximo de iterações i_{max} foi fixado em 1000 para as instâncias com menos de 100 clientes, 2000 para as instâncias com menos de 500 clientes e 3000 para as instâncias com mais de 500 clientes. O valor do segmento θ foi fixado em 200 iterações e o fator de reação η foi fixado em 0,7, definindo assim que os novos pesos corresponderão a 70% do desempenho do último segmento e 30% do valor do peso anterior. Os *scores* foram atualizados com σ_1 = 10, σ_2 = 5 e σ_3 = 2. O Algoritmo 1 mostra o pseudocódigo da implementação do ALNS.

Algoritmo 1 : Matheurística ALNS híbrida.

Inicialização: fixe todos os pesos em 1 e todos os *scores* em 0.
$s_{melhor} \leftarrow s \leftarrow s_{inicial}$, $T \leftarrow T_{inicial}$.
while $T > 0.01$ **do**
    $s' \leftarrow s$.
    escolha um operador de destruição e um de reparo usando uma roleta baseada nos pesos atuais. Aplique os operadores à s' e atualize o número de vezes que foram utilizados.
    resolva o subproblema (9)$-$(14), obtendo $z(s')$.
    if $z(s') > z(s)$ **then**
        $s \leftarrow s'$.
        if $z(s) > z(s_{melhor})$ **then**
            $s_{melhor} \leftarrow s$.
            atualize o *score* dos operadores utilizados com σ_1.
            aplique o procedimento de melhoria à s_{melhor}.
        else
            atualize o *score* dos operadores utilizados com σ_2.
        end if
    else
        if s' é aceita pelo critério do Simulated Annealing **then**
            $s \leftarrow s'$.
            atualize o *score* dos operadores utilizados com σ_3.
        end if
    end if
    if o número de iterações é múltiplo de θ **then**
        atualize os pesos de todos os operadores e reinicie seus *scores*.
        aplique o procedimento de melhoria à s_{melhor}.
    end if
    $T \leftarrow \phi T$.
end while
return s_{melhor}.

4. Resultados Computacionais

O algoritmo ALNS foi implementado em C++ e os subproblemas foram resolvidos pelo *software* de otimização comercial CPLEX Concert Technology versão 12.6. Todas as execuções foram realizadas em computadores equipados com processadores Intel Xeon^{TM} operando a 2,66 GHz. Na execução, o algoritmo ALNS utilizou menos de 1 GB de memória RAM. O modelo apresentado na Seção 2 foi implementado em CPLEX e executado em computadores com até 48 GB de memória RAM. Os computadores utilizados operavam sob o sistema operacional Scientific Linux 6.0.

Os ensaios computacionais foram realizados sobre um grande número de instâncias de teste, divididas em 3 grupos: 26 instâncias com 30 vértices, 24 instâncias com 324 vértices e 24 instâncias com 818 vértices. O número p de facilidades a serem instaladas variou de 2 a 50. O raio de atendimento S das facilidades instaladas foi fixado em 1,5 milhas para as instâncias com 30 vértices, 250 metros para as instâncias com 324 vértices e 750 metros para as instâncias com 818 vértices. Os dados das instâncias com 30 vértices foram propostas por Marianov e Serra (1998) e os das instâncias com 324 e 818 vértices foram propostas por Corrêa *et al.* (2008). Tais conjuntos de dados foram utilizados para avaliar a heurística MS proposta por Marianov e Serra (1998), o algoritmo genético construtivo AGC de Corrêa e Lorena (2006), o algoritmo de busca em agrupamentos de Corrêa *et al.* (2008) e a heurística de geração de colunas de Corrêa *et al.* (2009).

Os nomes das instâncias nas tabelas 1−3 contém os valores dos parâmetros utilizados na execução. Por exemplo, a instância 30_2_0_0_85 refere-se a um problema com 30 vértices, duas facilidades, o tipo de restrição probabilística (0 para tamanho de fila ou 1 para tempo de espera), o respectivo parâmetro de congestionamento (o número b de clientes na fila ou o tempo de espera τ em minutos) e a probabilidade mínima α, em porcentagem. A taxa de chegada μ foi fixada em 72 para a rede com 30 vértices e em 96 para as redes com 324 e 818 vértices. O parâmetro f_i das formulações (1)−(7), (9)−(14) e (15)−(22) foi calculado como fd_i, com f = 0,01 para as redes com 324 e 818 vértices. Para a rede com 30 vértices, f = 0,015 (restrições de tamanho da fila) ou f = 0,006 (restrições de tempo de espera).

As tabelas 1−3 apresentam, para cada instância, a melhor solução, a solução média (quando disponível) e o respectivo tempo computacional para seis métodos distintos. A melhor solução encontrada para cada instância está destacada em negrito. Na coluna CPLEX de cada tabela, a nova melhor solução obtida por um método exato aparece marcada com um asterisco. Para a heurística de geração de colunas, a coluna *Gap (%)* apresenta a diferença percentual entre a solução obtida e o limitante superior, ambos calculados pela heurística.

Na seção da heurística ALNS, cada instância foi executada 3 vezes. Na coluna *Média* pode-se comprovar a consistência do método proposto. Os valores apresentados nas colunas *Tempo (s)* também são valores médios.

Conforme apresentado na Tabela 1, a implementação do método exato provou a otimalidade para as 26 instâncias. O algoritmo ALNS atingiu a solução ótima em todas elas, demonstrando um desempenho melhor que os outros métodos da literatura, como pode ser comprovado na linha *Média* no final da tabela, sendo em média 10 vezes mais rápido que o CPLEX.

Na Tabela 2, usando a implementação do método exato, o CPLEX provou a otimalidade para 23 das 24 instâncias. o algortimo ALNS conseguiu encontrar a solução ótima para 19 instâncias e atingiu resultados melhores que os da literatura para as outras 5 instâncias.

Na Tabela 3, o CPLEX obteve a solução ótima para todas as instâncias. O algoritmo ALNS encontrou 19 delas e apresentou desempenho melhor em uma instância que o obtido anteriormente pela melhor heurística da literatura.

5. Conclusão

Este artigo apresentou um método híbrido para resolver o problema probabilístico de localização-alocação de máxima cobertura. O algoritmo é baseado numa heurística *adaptive large neighborhood search* utilizada para determinar as decisões de localização do problema. Um método

Tabela 1: Resultados computacionais para a instância com 30 vértices.

Instância	CPLEX			Heurística MS		AGC			Busca em Agrupamentos			Geração de Colunas			ALNS		
	Solução	Gap (%)	Tempo (s)	Solução	Tempo (s)	Melhor	Média	Tempo (s)	Melhor	Média	Tempo (s)	Solução	Gap (%)[1]	Tempo (s)	Melhor	Média	Tempo (s)
30_2_0_0_85	**3700**	0,000	0	**3700**	0,000	**3700**	3700	0,310	**3700**	3700,0	0,009	**3700**	0,000	0,530	**3700**	3700,0	12,333
30_2_0_1_85	**5100**	0,000	0	4630	0,000	5090	5090	0,360	5090	5090,0	0,018	5090	0,196	5,550	**5100**	5100,0	9,000
30_2_0_2_85	**5210**	0,000	0	4780	0,000	**5210**	5210	0,380	**5210**	5210,0	0,016	**5210**	1,321	4,770	**5210**	5210,0	4,667
30_2_0_2_95	**4520**	0,000	0	4470	0,000	**4520**	4513	0,300	**4520**	4520,0	0,015	**4520**	0,000	0,700	**4520**	4520,0	11,333
30_3_0_0_85	**5390**	0,000	0	5210	0,000	**5390**	5390	0,510	**5390**	5390,0	0,025	**5390**	0,119	28,030	**5390**	5390,0	5,000
30_3_0_1_85	**5390**	0,000	0	5210	0,000	**5390**	5390	0,480	**5390**	5390,0	0,024	**5390**	0,742	3,770	**5390**	5390,0	3,000
30_3_0_1_95	**5270**	0,000	6	5080	0,000	5240	5240	0,480	5240	5240,0	0,024	5240	0,763	14,750	**5270**	5270,0	44,667
30_3_0_2_85	**5390**	0,000	1	5210	0,000	**5390**	5390	0,470	**5390**	5390,0	0,023	**5390**	0,742	1,880	**5390**	5390,0	3,667
30_3_0_2_95	**5390**	0,000	0	5230	0,000	**5390**	5390	0,500	**5390**	5390,0	0,027	**5390**	0,622	11,450	**5390**	5390,0	3,667
30_3_1_49_90	**2160**	0,000	0	**2160**	0,000	**2160**	2160	0,530	**2160**	2160,0	0,009	**2160**	0,000	0,360	**2160**	2160,0	13,333
30_4_0_1_95	**5390**	0,000	0	5260	0,000	**5390**	5390	0,540	**5390**	5390,0	0,026	**5390**	1,391	30,880	**5390**	5390,0	8,333
30_4_1_42_85	**4600**	0,000	0	4550	0,000	**4600**	4600	0,610	**4600**	4600,0	0,022	**4600**	0,000	0,810	**4600**	4600,0	105,667
30_4_1_48_90	**1920**	0,000	0	1890	0,000	**1920**	1920	0,660	**1920**	1920,0	0,014	**1920**	0,000	0,440	**1920**	1920,0	17,333
30_4_1_49_90	**2880**	0,000	0	2870	0,000	**2880**	2877	0,590	**2880**	2880,0	0,013	**2880**	0,000	0,450	**2880**	2880,0	15,000
30_5_0_0_95	**5330**	0,000	13	5210	0,000	**5330**	5323	0,800	**5330**	5317,6	0,041	**5330**	0,375	6,910	**5330**	5330,0	288,667
30_5_1_40_85	**3050**	0,000	93	2910	0,000	3020	3001	0,740	**3050**	3033,2	0,021	**3050**	0,000	0,890	**3050**	3050,0	234,000
30_5_1_42_85	**5390**	0,000	1	5210	0,000	**5390**	5390	0,770	**5390**	5390,0	0,035	**5390**	0,464	34,170	**5390**	5390,0	15,000
30_5_1_48_90	**2400**	0,000	1	2280	0,000	2390	2390	0,740	**2400**	2398,8	0,019	**2400**	0,000	3,080	**2400**	2400,0	26,667
30_5_1_50_90	**4700**	0,000	1	4670	0,000	**4700**	4700	0,730	**4700**	4700,0	0,028	**4700**	0,000	0,830	**4700**	4700,0	42,667
30_6_0_0_95	**5410**	0,000	1	5390	0,000	**5410**	5392	0,840	**5410**	5391,0	0,037	**5410**	1,054	42,410	**5410**	5410,0	62,333
30_6_1_40_85	**3610**	0,605	10803	3480	0,000	**3610**	3610	0,910	**3610**	3608,8	0,031	**3610**	1,022	30,640	**3610**	3610,0	514,000
30_6_1_41_85	**5330**	0,188	10802	5120	0,000	**5300**	5274	1,000	**5330**	5286,4	0,038	**5300**	0,755	11,740	**5330**	5330,0	322,333
30_6_1_50_90	**5390**	0,000	1	5060	0,000	**5390**	5390	0,970	**5390**	5390,0	0,038	**5390**	0,395	48,440	**5390**	5390,0	31,667
30_7_1_40_85	**4060**	0,000	1	3860	0,000	**4060**	4060	1,030	**4060**	4060,0	0,039	**4060**	0,517	18,920	**4060**	4060,0	301,000
30_7_1_41_85	**5410**	0,000	1	5300	0,000	5390	5390	0,880	5390	5390,0	0,035	5350	2,159	41,840	**5410**	5410,0	61,333
30_8_1_41_85	**5470**	0,000	0	5390	0,000	**5470**	5470	0,950	**5470**	5470,0	0,032	**5470**	0,000	1,200	**5470**	5470,0	8,333
Média	**4533,1**	0,030	835,577	4389,6	0,000	4528,1	4525,0	0,657	4530,8	4527,1	0,025	4528,1	0,486	13,286	**4533,1**	4533,1	83,269

[1] gap da heurística

Tabela 2: Resultados computacionais para a instância com 324 vértices.

Instância	CPLEX			Heurística MS		AGC			Busca em Agrupamentos			Geração de Colunas			ALNS		
	Solução	Gap (%)	Tempo (s)	Solução	Tempo (s)	Melhor	Média	Tempo (s)	Melhor	Média	Tempo (s)	Solução	Gap (%)[1]	Tempo (s)	Melhor	Média	Tempo (s)
324_10_0_0_85	**37180**	0,000	13	37081	0,220	37145	37069,0	8,100	37173	37163,2	3,460	**37180**	0,000	275,110	**37180**	37180,0	1255,333
324_10_0_0_95	**21460**	0,000	9	21386	0,140	21431	21373,0	8,630	21455	21447,2	3,160	**21460**	0,000	65,200	**21460**	21460,0	865,667
324_10_0_1_85	**51000**	0,000	22	50750	0,200	50880	50711,0	7,690	50948	50946,3	3,410	**51000**	0,000	507,810	**51000**	51000,0	1435,000
324_10_0_1_95	**35360**	0,000	14	35250	0,160	35342	35304,0	7,790	35359	35354,6	3,170	**35360**	0,000	47,330	**35360**	35360,0	857,000
324_10_0_2_85	**59740**	0,000	22	59598	0,200	59624	59437,0	7,620	59693	59688,5	3,330	**59740**	0,000	604,480	**59740**	59740,0	987,667
324_10_0_2_95	**45390**	0,000	9	45300	0,200	45347	45245,0	7,810	45374	45354,6	3,100	**45390**	0,000	327,840	**45390**	45390,0	1292,000
324_10_1_40_85	**27700**	0,000	14	27583	0,170	27675	27602,0	8,540	27698	27692,1	3,370	**27700**	0,000	238,390	**27700**	27700,0	1406,333
324_10_1_41_85	**29360**	0,000	6	29288	0,140	29324	29260,0	8,270	29351	29341,9	3,520	**29360**	0,000	145,880	**29360**	29360,0	1182,667
324_10_1_42_85	**30950**	0,000	11	30902	0,170	30932	30895,0	8,340	30948	30943,3	3,330	**30950**	0,000	42,910	**30950**	30950,0	1034,667
324_10_1_48_90	**26920**	0,000	15	26855	0,140	26883	26835,0	8,350	26917	26910,6	3,500	**26920**	0,000	144,720	**26920**	26920,0	1492,667
324_10_1_49_90	**28330**	0,000	22	28206	0,250	28280	28221,0	8,280	28318	28310,3	3,430	**28330**	0,000	315,700	**28330**	28330,0	1747,000
324_10_1_50_90	**29680**	0,000	6	29638	0,220	29641	29593,0	8,260	29672	29665,5	3,360	**29680**	0,000	81,810	**29680**	29680,0	1127,333
324_20_0_0_85	**74360***	0,000	198	73981	0,830	73407	73001,0	23,690	74165	74106,7	9,710	74358	0,003	2630,440	**74360**	74357,7	6191,667
324_20_0_0_95	**42920**	0,000	22	42714	0,730	42577	42318,0	24,300	42840	42804,9	9,370	**42920**	0,000	2425,520	**42920**	42919,7	3936,333
324_20_0_1_85	**102000***	0,000	612	100628	1,020	99576	98353,0	24,690	101374	101177,4	9,070	101973	0,024	10801,330	101978	101974,0	7212,667
324_20_0_1_95	**70720**	0,000	55	70368	0,740	70471	70308,0	23,400	70656	70628,2	9,060	**70720**	0,000	1067,410	**70720**	70720,0	4417,333
324_20_0_2_85	**119470***	0,008	10804	118451	0,770	116639	115235,0	23,330	118771	118613,6	8,990	119448	0,019	10801,590	119455	119447,3	6388,000
324_20_0_2_95	**90780***	0,000	74	90424	0,810	89970	89355,0	24,150	90556	90521,2	9,400	**90780**	0,000	2369,980	**90780**	90780,0	6667,333
324_20_1_40_85	**55400***	0,000	52	55006	0,840	54804	54414,0	23,920	55306	55226,7	9,650	55396	0,007	8457,660	55399	55398,3	5488,667
324_20_1_41_85	**58720**	0,000	85	58577	1,020	58009	57571,0	24,700	58604	58583,4	10,330	**58720**	0,000	1377,420	**58720**	58720,0	7417,667
324_20_1_42_85	**61900**	0,000	35	61637	0,880	61545	61266,0	23,810	61847	61822,6	9,740	**61900**	0,000	735,140	**61900**	61900,0	3561,333
324_20_1_48_90	**53840***	0,000	49	53377	0,630	53300	52958,0	24,320	53793	53689,5	9,820	53838	0,004	4645,490	53839	53838,7	5912,667
324_20_1_49_90	**56660***	0,000	167	56180	1,340	56216	55813,0	24,150	56532	56429,9	9,550	56654	0,009	10800,920	56655	56655,0	7130,667
324_20_1_50_90	**59360***	0,000	56	59119	0,940	58941	58577,0	26,030	59285	59242,6	10,080	**59360**	0,000	1273,390	**59360**	59360,0	5551,333
Média	**52883,3**	0,000	515,500	52595,8	0,532	52415,0	52113,1	16,174	52776,5	52736,0	6,455	52880,7	0,003	2507,645	**52881,5**	52880,9	3523,292

* nova melhor solução obtida por um método exato
[1] gap da heurística

Tabela 3: Resultados computacionais para a instância com 818 vértices.

Instância	CPLEX			Heurística MS		AGC			Busca em Agrupamentos			Geração de Colunas			ALNS		
	Solução	Gap (%)	Tempo (s)	Solução	Tempo (s)	Melhor	Média	Tempo (s)	Melhor	Média	Tempo (s)	Solução	Gap (%)[1]	Tempo (s)	Melhor	Média	Tempo (s)
818_10_0_0_95	**21460**	0,000	557	21429	4,940	21455	21449,3	43,650	**21460**	21459,9	11,230	**21460**	0,000	163,08	**21460**	21460,0	6818,000
818_10_0_1_95	**35360**	0,000	657	35339	13,050	35356	35346,0	50,450	**35360**	35360,0	11,540	**35360**	0,000	207,69	**35360**	35360,0	10407,000
818_10_0_2_95	**45390**	0,000	648	45375	12,090	45387	45377,4	49,370	**45390**	45390,0	12,170	**45390**	0,000	251,33	**45390**	45390,0	7400,000
818_10_1_48_90	**26920**	0,000	586	26907	11,380	26915	26901,0	48,350	**26920**	26920,0	11,320	**26920**	0,000	185,41	**26920**	26920,0	8212,000
818_10_1_49_90	**28330**	0,000	659	28309	8,630	28320	28298,0	47,780	**28330**	28330,0	12,230	**28330**	0,000	196,06	**28330**	28330,0	8052,667
818_10_1_50_90	**29680**	0,000	682	29661	18,280	29678	29673,8	47,810	**29680**	29680,0	11,330	**29680**	0,000	199,28	**29680**	29680,0	9078,333
818_20_0_0_85	**74360**	0,000	785	74313	75,050	74341	74279,4	90,040	**74360**	74359,8	66,810	**74360**	0,000	412,74	**74360**	74360,0	17183,667
818_20_0_0_95	**42920**	0,000	562	42793	25,670	42870	42817,7	76,200	**42920**	42918,9	54,840	**42920**	0,000	248,13	**42920**	42920,0	9890,333
818_20_0_1_85	**102000***	0,000	1491	101933	76,060	101955	101898,8	89,190	**102000**	101998,9	67,360	**102000**	0,000	447,06	**102000**	102000,0	16602,000
818_20_0_1_95	**70720**	0,000	610	70644	37,030	70653	70557,1	80,420	**70720**	70719,2	62,910	**70720**	0,000	390,03	**70720**	70720,0	16880,667
818_20_0_2_85	**119480***	0,000	1579	119397	105,480	119445	119306,4	89,180	**119480**	119477,6	74,360	**119480**	0,000	473,52	**119480**	119480,0	19074,667
818_20_0_2_95	**90780**	0,000	841	90730	75,190	90747	90672,9	84,990	**90780**	90779,6	66,800	**90780**	0,000	460,19	**90780**	90780,0	14622,667
818_20_1_40_85	**55400***	0,000	833	55325	66,110	55341	55235,7	88,330	**55400**	55399,0	61,040	**55400**	0,000	293,62	**55400**	55400,0	15044,333
818_20_1_41_85	**58720***	0,000	641	58637	69,630	58706	58677,6	87,620	**58720**	58719,9	57,310	**58720**	0,000	372,28	**58720**	58720,0	14384,000
818_20_1_42_85	**61900***	0,000	929	61814	71,810	61839	61719,7	90,430	**61900**	61898,8	58,710	**61900**	0,000	380,64	**61900**	61900,0	14487,000
818_20_1_48_90	**53840**	0,000	640	53728	22,410	53814	53762,2	87,870	**53840**	53839,6	59,560	**53840**	0,000	343,08	**53840**	53840,0	13808,667
818_20_1_49_90	**56660**	0,000	948	56602	34,340	56605	56465,3	88,710	**56660**	56660,0	67,420	**56660**	0,000	324,64	**56660**	56660,0	13008,333
818_20_1_50_90	**59360**	0,000	877	59269	39,840	59336	59279,8	87,040	**59360**	59359,9	58,480	**59360**	0,000	364,84	**59360**	59360,0	14568,667
818_50_0_0_85	**185900***	0,000	2956	185426	756,720	184428	184153,6	177,340	185880	185775,6	364,200	185898	0,001	3695,94	185637	185587,7	24094,333
818_50_0_1_85	**255000***	0,000	6332	254509	730,200	253438	252884,6	171,860	254985	254905,0	387,370	**255000**	0,000	5072,75	254996	254987,3	23978,000
818_50_0_2_85	**298700***	0,000	4435	298217	757,730	296763	296182,8	171,740	298582	298517,6	392,290	298490	0,070	6004,53	298692	298648,3	23729,667
818_50_1_48_90	**134600***	0,000	785	134088	596,110	134079	133999,4	177,670	134598	134561,6	335,000	**134600**	0,000	2317,55	134597	134593,3	25037,333
818_50_1_49_90	**141650***	0,000	1719	141281	571,170	140439	140143,4	174,960	141586	141532,8	356,800	**141650**	0,000	2713,86	**141650**	141606,7	24425,000
818_50_1_50_90	**148400***	0,000	1946	147931	667,670	147202	147123,8	173,960	148383	148321,9	337,400	**148400**	0,000	2444,64	148397	148378,0	25031,667
Média	**91563,8**	0,000	1362,417	91402,4	201,941	91213,0	91091,9	98,957	91553,9	91536,9	124,937	91554,9	0,003	1165,120	91552,0	91545,1	15659,125

* nova melhor solução obtida por um método exato
[1] gap da heurística

exato de programação matemática é utilizado para determinar a solução dos subproblemas de alocação. As melhores soluções obtidas pela metaheurística podem ser aperfeiçoadas por um procedimento flexível de melhoria. O método híbrido apresentou um desempenho consistente, encontrando 10 soluções melhores que as da literatura, sendo 4 delas ótimas. O critério de parada exclusivamente baseado em *Simulated Annealing* contribui para o aumento dos tempos computacionais nas instâncias maiores.

6. Agradecimentos

O primeiro autor agradece ao CIRRELT – Centre Interuniversitaire de Recherche sur les Réseaux d'Enterprise, la Logistique et le Transport, ao Department of Operations and Decision Systems e à Faculty of Administration Sciences da Université Laval pelo apoio financeiro.

Referências

Arakaki, R. G. I. e Lorena, L. A. N., A constructive genetic algorithm for the maximal covering location problem, in *Proceedings of the 4th Metaheuristics International Conference*, Porto, Portugal, 2001.

Barrena, E., Ortiz, D. C., Coelho L. C. e Laporte, G., A fast and efficient adaptive large neighborhood search heuristic for the passenger train timetabling problem with dynamic demand, *Technical Report 64*, CIRRELT, Montréal, 2013.

Bartodziej, P., Derigs, U., Malcherek, D. e Vogel, U. (2009), Models and algorithms for solving combined vehicle and crew scheduling problems with rest constraints: an application to road feeder service planning in air cargo transportation, *OR Spectrum*, 31, 405-429.

Berman, O., Drezner, Z. e Wesolowsky, G. O. (2009), The maximal covering problem with negative weights, *Geographical Analysis*, 41, 30-42.

Berman, O., Hajizadeh, I. e Krass, D. (2013), The maximum covering problem with travel time uncertainty, *IIE Transactions,*, 45, 81-96.

Church, R. e ReVelle, C. (1974), The maximal covering location problem, *Papers in Regional Science*, 32, 101-118.

Coelho, L. C., Cordeau, J.-F. e Laporte, G. (2012a), The inventory-routing problem with transshipment, *Computers & Operations Research*, 39, 2537-2548.

Coelho, L. C., Cordeau, J.-F. e Laporte, G. (2012b), Consistency in multi-vehicle inventory-routing, *ransportation Research Part C: Emerging Technologies*, 24, 270-287.

Corrêa, F. A. e Lorena, L. A. N., Using the constructive genetic algorithm for solving the probabilistic maximal covering location-allocation problem, in *Proceedings of the I Workshop on Computational Intelligence*, Ribeirão Preto, Brasil, 2006.

Corrêa, F. A., Chaves, A. A. e Lorena, L. A. N. (2008), Hybrid heuristics for the probabilistic maximal covering location-allocation problem, *Operational Research. An International Journal*, 7, 323-344.

Corrêa, F. A., Lorena, L. A. N. e Ribeiro, G. M. (2009), A decomposition approach for the probabilistic maximal covering location-allocation problem, *Computers & Operations Research*, 36, 2729-2739.

Daskin, M. S., *Network and Discrete Location: Models, Algorithms and Applications*, John Wiley & Sons, New York, 1995.

Galvão, R. D. (2004), Uncapacitated facility location problems: contributions, *Pesquisa Operacional*, 24, 7-38.

Galvão, R. D. e ReVelle, C. (1996), A lagrangean heuristic for the maximal covering location problem, *European Journal of Operational Research*, 88, 114-123.

Hale, T. S. e Moberg, C. R. (2003), Location science research: a review, *Annals of Operations Research*, 123, 21-35.

Hemmelmayr, V. C., Cordeau, J.-F. e Crainic, T. G. (2012), An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics, *Computers & Operations Research*, 39, 3215-3228.

Lorena, L. A. N. e Pereira, M. A. (2002), A lagrangean/surrogate heuristic for the maximal covering location problem using Hillsman's edition, *International Journal of Industrial Engineering*, 9, 57-67.

Maniezzo, V., Stützle T. e Voß, S., *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*, Springer, New York, 2009.

Marianov, V. e Serra, D. (1998), Probabilistic maximal covering location-allocation models for congested systems, *Journal of Regional Science*, 38, 401-424.

Pepin, A,-S., Desaulniers, G. Hertz, A. e Huisman, D. (2009), A comparison of five heuristics for the multiple depot vehicle scheduling problem, *Journal of Scheduling*, 12, 17-30.

Pereira, M. A., Lorena, L. A. N. e Senne, E. L. F. (2007), A column generation approach for the maximal covering location problem, *International Transactions in Operations Research*, 14, 349-364.

Pirkul, H. e Schilling, D. A. (1991), The maximal covering location problem with capacities on total worload, *Management Science*, 37, 233-248.

Ropke, S. e Pisinger, D. (2006), An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows, *Transportation Science*, 40, 455-472.

Ropke, S. e Pisinger, D. (2006), A unified heuristic for a large class of vehicle routing problems with backhauls, *European Journal of Operational Research*, 171, 750-755.

Serra, D. e Marianov, V., New trends in public facility location modeling, *Technical Report 755*, UFP Economics and Business, Barcelona, 2004.