Uma Abordagem de Recozimento Simulado com Busca Local para o Problema Integrado de Localização e Roteamento

Kamyla Maria Ferreira
Unidade de Matemática e Tecnologia - UFG/Regional Catalão, 75704-020, Catalão-GO, Brasil
Kamylamaaria@gmail.com

Thiago Alvez de Queiroz
Unidade de Matemática e Tecnologia - UFG/Regional Catalão, 75704-020, Catalão-GO, Brasil
taq@ufg.br

RESUMO

Este trabalho apresenta uma heurística baseada no método de Recozimento Simulado combinada com uma busca local para o problema integrado de localização de instalações com roteamento de veículos. Considera-se a versão capacitada do problema, em que os depósitos e veículos são limitados. A heurística desenvolvida trabalha sobre o Recozimento Simulado que, claramente, aceita soluções ruins, além de agregar operadores de vizinhança, uma busca local, uma função de diversificação e uma função de perturbação. A heurística é testada em três conjuntos de instâncias bem conhecidas da literatura, em que os resultados foram comparados com as melhores estratégias apresentadas pela literatura. Os experimentos computacionais indicam que a heurística desenvolvida é competitiva em comparação com a literatura recente do problema.

PALAVRAS CHAVE. Problema de Localização e Roteamento, Recozimento Simulado, Busca Local.

Área Principal: Logística e Transportes.

ABSTRACT

This work presents a heuristic based on the Simulated Annealing method combined with a local search to the integrated facility location and vehicle routing problem. The capacitated version of this problem is solved, that is, depots and vehicles are limited by a given capacity. The heuristic proposed considers the Simulated Annealing that, of course, accepts bad solutions, and also has neighborhood operations, a local search routine, a diversification criteria and a perturbation routine. This heuristic is applied over three set of well-known instances from the literature, where solutions are compared with the best known solutions from the literature. The computational experiments indicate that the heuristic is competitive when comparing with the recent literature of this problem.

KEYWORDS. Location-Routing Problem, Simulated Annealing, Local Search.

Main Area: Logistics and Transport.

1. Introdução

Diversas empresas buscam otimizar os custos logísticos, já que estes correspondem a uma grande parte de seus gastos (Barreto, 2004). Para tanto, é necessário implementar um planejamento logístico, que em alguns casos consiste em tomar decisões para reduzir as despesas de localização de depósitos e, posteriormente, das rotas de distribuição partindo destes depósitos. O problema que agrega as decisões simultâneas de localizar e rotear é referenciado na literatura como Problema de Localização e Roteamento – PLR (Laporte e Norbert, 1981).

Em linhas gerais, o PLR busca determinar a localização das instalações (depósitos, armazéns, fábricas, etc.) e traçar rotas que, partindo delas, atendem clientes dispersos, com objetivo de minimizar o custo global das decisões de localizar e rotear (Prodhon e Prins, 2014).

O PRL é classificado como um problema NP-difícil, uma vez que generaliza o problema de Localização de Instalações e o de Roteamento de Veículos, que são ambos NP-difíceis (Garey e Johnson, 1979). Sendo assim, a literatura tem buscado desenvolver mais métodos heurísticos do que exatos para resolvê-lo. Um apanhado geral sobre o PLR foi feito recentemente por Prodhon e Prins (2014), enquanto suas variantes foram sumarizadas em Drexl e Schneider (2015).

Os métodos exatos geram a solução ótima, entretanto o tempo computacional tem sido exponencial no tamanho da entrada, de forma que não conseguem lidar com instâncias grandes. Laporte e Norbert (1981) formalizaram o PLR e propuseram um algoritmo *branch-and-bound*. Uma das estratégias exatas mais eficientes é o algoritmo *branch-and-cut-and-price* de Contardo et al. (2014a).

No que tange aos métodos heurísticos para o PLR, que computam uma solução boa (em alguns casos a ótima), dentro de um tempo computacional aceitável para a tomada de decisões rápidas, destacam-se: Tuzun e Burke (1999), que propuseram uma abordagem de duas fases com busca Tabu; Yu et al. (2010), que desenvolveram um método que parte do Recozimento Simulado (*Simulated Annealing*) combinado com operadores de vizinhança e uma busca local; Jarboui et al. (2013), que propuseram algoritmos de Busca em Vizinhança Variável; Contardo et al. (2014b), que propuseram uma heurística híbrida envolvendo GRASP e a resolução de modelos de programação inteira, conseguindo melhorar a solução de várias instâncias para as quais a solução ótima ainda é desconhecida.

O objetivo deste trabalho é apresentar uma heurística para o PRL partindo de algumas das abordagens da literatura. Usou-se a mesma estrutura da heurística de Yu et al. (2010), a partir da qual se propôs novos operadores de vizinhança e uma nova rotina de busca local, além de integrar funções para escapar de ótimos locais. Mais detalhes são dados na próxima seção.

Os experimentos computacionais são discutidos na Seção 3. Os resultados foram satisfatórios, uma vez que se conseguiu obter soluções próximas das apresentadas na literatura recente do PLR. Por fim, conclusões e direções para trabalhos futuros são dadas na Seção 4.

2. Caracterização da Heurística

No PLR é dado um grafo não-orientado $G = (V,E)$, sendo V o conjunto de vértices, que contém todos os possíveis locais para abrir depósitos em $I = \{1,2,\ldots,m\}$ e os clientes $J = \{m+1,m+2,\ldots m+n\}$, e E sendo o conjunto de arestas que ligam dois nós de V, com exceção de arestas para conectar depósito com depósito, tal que cada $e \in E$ tem um custo de travessia c_e. Além disso, cada $i \in I$ tem capacidade W_i e custo de abertura O_i, enquanto cada $j \in J$ tem demanda d_j e deve ser visitado exatamente uma vez. Também, há um conjunto de K veículos idênticos, cada um com capacidade Q e custo de utilização F_i, que é variável de acordo com o depósito que está servindo. No PLR, cada veículo executa exatamente uma rota, que inicia e finaliza no mesmo depósito, tal que a sua capacidade Q deve ser respeitada. Um depósito pode atender uma ou mais rotas conforme a sua capacidade W_i, tal que toda a demanda dos clientes deve ser suprida. O objetivo é minimizar o custo global associado a abertura de depósitos e das rotas que deles partem.

A heurística usada para resolver o PRL parte do Recozimento Simulado, com a agregação de novas rotinas para explorar eficientemente o espaço de soluções. A estrutura geral da heurística

implementada neste trabalho é apresentada no Algoritmo 1. Os parâmetros de entrada são: I_{iter}, T_0, T_f, K, P, NN e α. O I_{iter} é o número de iterações locais realizadas caso a temperatura T, inciando em T_0, ainda não alcance seu valor final T_f. Ademais, K é a constante de *Boltzmann* utilizada na função de aceitação de soluções ruins, P é a penalidade aplicada a cada depósito cuja capacidade é violada, NN é o limite de reduções consecutivas que T pode ter e α é o coeficiente de arrefecimento da temperatura. O valor de Δ corresponde à diferença entre o valor das soluções Y e X.

Algorithm 1: Estrutura geral da heurística para o PLR.

Seja X uma **solução inicial**;
$T \leftarrow T0$; $I \leftarrow 0$; $nn \leftarrow 0$; $X_{best} \leftarrow X$;
repita
  repita
    $Y \leftarrow$ Escolha com igual probabilidade e aplique um dos **Operadores de Vizinhança** em X;
    $\Delta \leftarrow custo(Y,P) - custo(X,P)$;
    se $\Delta \leq 0$ **então** $X \leftarrow Y$;
    else
      se $rand(0,1) < \exp\frac{-\Delta}{K \cdot T}$ **então** $X \leftarrow Y$;
    se $custo(X,P) < custo(X_{best},P)$ **então** $X_{best} \leftarrow Y$; $nn \leftarrow 0$;
    $I \leftarrow I + 1$;
  até $I \leq I_{iter}$;
  $T \leftarrow \alpha T$; $I \leftarrow 0$; $nn \leftarrow nn + 1$;
  $Y \leftarrow$ Aplique a rotina de **Busca local** em $Xbest$;
  se $custo(Y,P) < custo(X_{best},P)$ **então** $X_{best} \leftarrow Y$;
  $X \leftarrow X_{best}$;
  se $nn = 20$ ***OU*** $nn = 40$ ***OU*** $nn = 60$ ***OU*** $nn = 80$ **então**
    $X \leftarrow$ Aplique a função de **Diversificação** em X;
  se $nn > 80$ **então**
    se $(80 \mod 2) \neq 0$ **então**
      $X \leftarrow$ Aplique a função de **Perturbação** em X;
    else $X \leftarrow$ Aplique a função de **Diversificação** em X;
  se $custo(X,P) < custo(X_{best},P)$ **então** $X_{best} \leftarrow X$;
até $T \leq T_f$ ***E*** $nn \neq NN$;
retorna *solução em* X_{best};

As rotinas que fazem parte do Algoritmo 1 são discutidas adiante. Uma solução é representada por um vetor contendo uma sequência de números que são: os depósitos em I, todos os clientes em J e zeros para indicar o fim de rotas, de forma a não ultrapassar a capacidade do veículo. Assim, este vetor começa sempre por um depósito e caso esteja aberto, vem seguido por um ou mais clientes podendo ter zeros entre eles, caso contrário, vem seguido por dois zeros consecutivos. Vale destacar que não se aceita que um veículo tenha a sua capacidade de carga ultrapassada, porém um depósito pode tê-la. Assim, o custo da solução consiste na soma: custo dos depósitos abertos, custo de atravessar as arestas e mais a penalidade P para cada depósito com capacidade violada.

2.1. Solução Inicial

O procedimento para gerar a solução inicial é guloso e consiste de: (i) escolhe-se de forma aleatória um $i \in J$ para ser depósito; (ii) adicionam-se clientes j em i respeitando sempre o menor valor de c_{ij}, tal que são adicionados clientes em i até o limite de sua capacidade W_i; e, (iii) atingido o limite de i, volta-se ao passo (i) até que todos os clientes tenham sido atribuídos a algum depósito em I.

Após o procedimento guloso que atribui clientes a depósitos, monta-se o vetor solução originando as rotas para cada depósito aberto. Partindo do primeiro depósito em I, coloca-o na primeira posição do vetor e se ele estiver fechado, dois zeros são adicionados e parte-se para o próximo depósito em I. Caso um $i \in I$ esteja aberto, os seus clientes vão sendo inseridos no vetor de acordo com a ordem em que foram atribuídos pelo procedimento guloso. Ao chegar no primeiro cliente $j \in J$, em que a soma das demandas dos clientes tenha ultrapassado a capacidade Q do veículo, um zero é adicionado antes de j para indicar o fim da rota e que j é o primeiro cliente da próxima rota partindo de i. Esse procedimento continua até que todos os depósitos e clientes tenham sido posicionados no vetor solução.

2.2. Operadores de Vizinhança

A heurística é composta por sete operadores de vizinhança que basicamente fazem mudanças na posição dos elementos que estão no vetor solução. Em resumo, os operadores são:

- Inserção: insere um elemento de um posição i antes de outro na posição j, ambas escolhidas de forma aleatória. Esta operação envolve uma de quatro opções, quais sejam: (a) i e j são ambos depósitos, tal que consiste em fechar i e transferir os seus clientes para o depósito que precede j; (b) i e j são ambos clientes, tal que i vem imediatamente antes de j; (c) i é depósito e j é cliente, tal que i passará a atender um novo conjunto de clientes iniciando em j e os antigos clientes de i passam a ser atendidos pelo depósito que precede i; e, (d) i é cliente e j é depósito, tal que i é simplesmente inserido antes de j;
- Troca: realiza a troca dos elementos que estão nas posições i e j escolhidas aleatoriamente. Surgem as mesmas quatro opções da Inserção;
- Inversão: os elementos que estão entre as posições de i a j escolhidas aleatoriamente são considerados em ordem inversa dentro vetor;
- Inserção de Uma Sequência: insere uma sequência de tamanho aleatório limitado a metade do vetor solução antes de uma posição também escolhida aleatoriamente e fora da sequência;
- Inserção de Duas Sequências: considera duas sequências de tamanho aleatório cujos tamanhos são limitados a metade do vetor solução. A primeira sequência é divida ao meio, sendo a sua primeira metade inserida na posição imediatamente antes do início da segunda e a sua outra metade inserida imediatamente após a posição final da segunda sequência;
- Troca de Zeros: realiza a troca de posição dos elementos zero com o intuito de modificar todas as rotas. O primeiro zero assume a primeira posição anterior a sua (faz-se uma troca com o elemento daquela posição), o segundo zero vai para a primeira posição posterior a sua, o terceiro zero vai para a primeira posição anterior a sua e, assim por diante, seguindo esta alternância;
- Troca de Duas Sequências: considera duas sequências de tamanho aleatório cujos tamanhos são limitados a metade do vetor solução. O movimento consiste em trocar as duas sequências de posição;

2.3. Busca Local

A busca local foi dividida em duas operações determinísticas, uma de troca e outra de inserção, que são aplicadas sequencialmente. Primeiro, faz-se a troca, que consiste no operador de Troca considerando todos os pares $\{i, j\}$ de posição no vetor solução. A operação que resulta na solução de menor custo é efetivamente aplicada ao fim.

Similarmente, a operação de inserção na busca local consiste em aplicar o operador de Inserção tomando todos os pares de elementos do vetor solução. Assim, uma das inserções, aquela que resultar na solução de menor custo, é efetivamente considerada.

2.4. Função de Diversificação

Tentando estabelecer uma rotina que possa diversificar satisfatoriamente a solução dada como entrada, a função de Diversificação consiste na aplicação da operação de Troca Inversa seguida pela Troca de Zeros, as quais são aplicadas consecutivamente. Além disso, a solução da primeira operação é usada como entrada para a segunda, de forma que a função retorna a solução da segunda operação.

A Troca Inversa funciona basicamente igual a operação de Troca de Duas Sequências, porém com a diferença de que as sequências são assumidas em ordem inversa.

A partir de testes preliminares, escolheu-se aplicar a função de diversificação a cada ciclo de 20 iterações consecutivas em que não há melhora no valor da melhor solução. A ideia é diversificar a exploração do espaço de busca, ao mesmo tempo que busca escapar de ótimos locais.

2.5. Função de Perturbação

A função de Perturbação é aplicada após 80 iterações consecutivas sem melhora, com o mesmo intuito da Diversificação. Ela opera efetuando a troca de elementos de três rotas, para as quais é calculado o centroide da rota, isto é, o seu centro geométrico. Estratégia similar foi adotada por Escobar et al. (2013).

A primeira rota, r_1, é escolhida aleatoriamente do vetor solução. A segunda rota, r_2, é escolhida para ser aquela de menor distância com r_1, enquanto r_3 tem a menor distância com relação a r_2. A menor distância Euclidiana é tomada com relação ao centroide das rotas.

Esta função aplica dois movimentos consecutivos considerando todo cliente j_1 de r_1, todo cliente j_2 de r_2 e aresta $\{i_2, f_2\}$ de r_2, com $j_2 \neq i_2$ e $j_2 \neq f_2$, e toda aresta $\{i_3, f_3\}$ de r_3. Os movimentos são: (i) j_1 é inserido entre os vértices da aresta $\{i_2, f_2\}$; e, (ii) j_2 é inserido entre os vértices da aresta $\{i_3, f_3\}$. A nova solução é aquela de menor custo entre todas as originadas a partir da aplicação de (i) e (ii) nestas três rotas.

3. Experimentos Computacionais

A heurística e suas rotinas foram codificados na linguagem C e os experimentos ocorreram em um computador com processador Intel Core i7-4790K de 4.0 GHz, 32 GB de memória RAM e sistema operacional Linux, rodando em uma única *thread*. Os testes ocorreram em 80 instâncias, sendo as 14 de Barreto (2004), as 30 de Prins et al. (2006) e as 36 de Tuzun e Burke (1999), consideradas padrão para o PLR, em que várias delas ainda não possuem o valor ótimo conhecido.

Os parâmetros iniciais da heurística influenciam na sua convergência, por isso foram calibrados por meio do pacote *irace* (*Iterated Race for Automatic Algorithm Configuration*) de (Lopez-Ibanez et al., 2011). Somente o valor de *NN* que foi tomado fixo em 100, pois está ligado com a proposta das funções de Diversificação e Pertubação. Após a calibragem, os seguintes parâmetros foram adotados: $I_{iter} = 5132$, $K = 0,26$, $T_0 = 31$, $T_f = 0,46$, $\alpha = 0.99$, $P = \frac{2\sum_{i \in I} O_i}{m}$ e semente igual a 3229.

As Tabelas de 1 a 3 apresentam os resultados encontrados pela heurística dada no Algoritmo 1 para cada grupo de instância. Cada linha dessas tabelas traz: nome da instância, seguido pelo valor de n e m; número de veículos utilizados na solução; tempo computacional gasto em segundos; valor da solução encontrada; valor da melhor solução conhecida[1]; e, a diferença relativa (GAP em porcentagem) entre a solução da heurística e a melhor conhecida.

Os valores nessas tabelas foram obtidos para uma única execução da heurística dado o valor de semente após a calibragem. Os valores em negrito indicam que o GAP é zero. Comenta-se ainda que a comparação de tempo não foi feita com a literatura devido as diferenças nas máquinas usadas nos experimentos e pelo fato de alguns autores não terem informado com precisão o tempo gasto.

Os resultados para as instâncias de Barreto (2004) estão na Tabela 1, com número de clientes variando entre 21 a 150 e depósitos entre 5 a 10. A heurística conseguiu soluções iguais as

[1] O melhor valor entre aqueles retornados por Yu et al. (2010), Hemmelmayr et al. (2012) e Contardo et al. (2014b).

melhores conhecidas para 6 das 14 instâncias. Na média, o GAP foi de $1,896\%$, com pior resultado de $6,483\%$ para a instância $Min134(134x5)$. O tempo computacional médio foi de $1.116,57$ segundos.

Tabela 1: Resultado para as instâncias de Barreto (2004).

Instância (n x m)	#Veículos	Tempo (s)	Valor da Solução	Melhor Solução Conhecida	GAP (%)
Christ50 (50x5)	6	489	565,6	565,6	**0,000**
Christ75 (50x5)	9	1.660	871,3	844,4	3,186
Christ100 (100x10)	8	1.867	878,96	833,4	5,467
Das88 (88x8)	6	1807	365,8	355,8	2,810
Das150 (150x10)	11	4.899	45.645,5	43.919,9	3,929
Gaspelle (21x5)	4	76	424,9	424,9	**0,000**
Gaspelle2 (22x5)	3	82	585,1	585,1	**0,000**
Gaspelle3 (29x5)	4	120	515,1	512,1	0,586
Gaspelle4 (32x5)	4	287	562,2	562,2	**0,000**
Gaspelle5 (32x5)	3	235	504,3	504,3	**0,000**
Gaspelle6 (36x5)	4	298	468,8	460,4	1,824
Min27 (27x5)	4	94	3.062	3.062	**0,000**
Min134 (134x5)	10	2.322	6.079,1	5709	6,483
Or117 (117x14)	5	1.396	12.568,9	12.290,3	2,267

Tabela 2: Resultado para as instâncias Prins et al. (2006).

Instância (n x m)	#Veículos	Tempo (s)	Valor da Solução	Melhor Solução Conhecida	GAP (%)
20-5-1 (20x5)	5	44	54.793	54.793	**0,000**
20-5-1b (20x5)	3	201	39.104	39.104	**0,000**
20-5-2 (20x5)	5	119	48.908	48.908	**0,000**
20-5-2b (20x5)	3	100	37.806	37.542	0,703
50-5-1 (50x5)	12	1.743	90.111	90.111	**0,000**
50-5-1b (50x5)	6	1.385	63.242	63.242	**0,000**
50-5-2 (50x5)	12	699	88.978	88.298	0,770
50-5-2b (50x5)	6	1.341	68.348	67.308	1,545
50-5-2bBIS (50x5)	6	449	52.270	51.822	0,864
50-5-2BIS (50x5)	12	567	84.912	84.055	1,019
50-5-3 (50x5)	12	1.698	88.538	86.203	2,709
50-5-3b (50x5)	6	1.075	62.880	61.830	1,698
100-5-1 (100x5)	24	2.830	279.174	274.814	1,586
100-5-1b (100x5)	12	5.369	219.013	213.615	2,527
100-5-2 (100x5)	24	2.219	196.967	193.671	1,702
100-5-2b (100x5)	11	2.861	158.270	157.095	0,748
100-5-3 (100x5)	24	5717	202.095	200.079	1,008
100-5-3b (100x5)	11	2.486	163.830	152.441	7,471
100-10-1 (100x10)	25	1.968	323.423	287.983	12,306
100-10-1b (100x10)	12	1.923	272.823	231.763	17,716
100-10-2 (100x10)	23	3.903	246.890	243.590	1,355
100-10-2b (100x10)	11	4.798	207.816	203.988	1,876
100-10-3 (100x10)	24	3.139	255.905	250.882	2,002
100-10-3b (100x10)	11	3.329	244.849	204.317	19,838
200-10-1 (200x10)	47	19.908	484.964	475.327	2,027
200-10-1b (200x10)	22	6.510	404.426	377.327	7,182
200-10-2 (200x10)	48	9.045	452.985	449.291	0,822
200-10-2b (200x10)	23	4.892	487.534	374.330	30,242
200-10-3 (200x10)	46	6.006	482.634	469.433	2,812
200-10-3b (200x10)	22	4.272	369.924	362.817	1,959

A Tabela 2 traz os resultados para as instâncias de Prins et al. (2006), com número de clientes variando de 20 a 200 e depósitos entre 5 a 10. Observa-se que cinco valores coincidiram com

a melhor solução conhecida e o GAP médio foi de $4,149\%$. Com relação ao tempo computacional, uma instância precisou de quase 6 horas de execução, enquanto as demais tiveram valores variando entre 44 e 9.045 segundos. É importante destacar que o PLR envolve decisões de nível estratégico (localizar depósitos) e operacional (determinar rotas), podendo justificar o alto tempo de execução.

A Tabela 3 sumariza os resultados para as instâncias de Tuzun e Burke (1999), que possuem a maior quantidade de clientes e depósitos. Para esse grupo de instâncias, a heurística obteve valores próximos a melhor solução conhecida com GAP médio de $3,664\%$, sendo que apenas duas instâncias apresentou um GAP superior a 10%. O tempo computacional foi um pouco maior do que aquele requerido pelas instâncias de Prins et al. (2006), chegando no pior caso a sete horas de execução para a instância $123112(200x10)$.

Tabela 3: Resultado para as instâncias de Tuzun e Burke (1999).

Instância (n x m)	#Veículos	Tempo (s)	Valor da Solução	Melhor Solução Conhecida	GAP (%)
111112 (100x10)	11	7486	1.541,6	1.467,68	5,037
111122 (100x20)	11	5704	1.505,6	1.449,2	3,892
111212(100x10)	10	4117	1.426,5	1.394,8	2,273
111222(100x20)	11	5031	1.470,7	1.432,29	2,682
112112(100x10)	11	5535	1.176,2	1.167,16	0,775
112122(100x20)	11	4886	1.123,9	1.102,24	1,965
112212 (100x10)	11	4212	824,3	791,66	4,123
112222 (100x20)	11	5562	731	728,3	0,371
113112 (100x10)	11	4369	1.250,1	1.238,49	0,937
113122(100x20)	11	1525	1.276,5	1.245,31	2,504
113212 (100x10)	11	7288	921,3	902,26	2,110
113222 (100x20)	11	7989	1.033,6	1.018,29	1,503
121112 (200x10)	21	13382	2.388,9	2.243,4	6,486
121122 (200x20)	21	16158	2.359,4	2.138,4	10,335
121212 (200x10)	21	18478	2.310,7	2.209,3	4,59
121222 (200x20)	21	18124	2.364,7	2.225,1	6,274
122112 (200x10)	21	11429	2.157,3	2.073,7	4,031
122122 (200x20)	20	11084	1.853,9	1.692,2	9,556
122212 (200x10)	20	10196	1.458,99	1.453,2	0,398
122222 (200x20)	22	11269	1.103,2	1.082,7	1,893
123112 (200x10)	21	25218	2.034,4	1.954,7	4,077
123122 (200x20)	20	19501	1.976,9	1.918,9	3,022
123212 (200x10)	21	13324	1.782,4	1.762	1,158
123222 (200x20)	22	12352	1.460,5	1.391,7	4,943
131112 (300x10)	16	5900	1.988	1.866,8	6,492
131122 (300x20)	16	6551	1.920,5	1.823,5	5,319
131212(300x20)	16	4941	2.031,4	1.964,3	3,416
131222 (300x20)	16	6426	1.905	1.792,8	6,258
132112 (300x10)	16	7853	1.475,5	1.443,3	2,231
132122 (300x20)	15	7720	1.437,6	1.434,6	0,209
132212 (300x10)	17	15190	1.211,7	1.204,4	0,606
132222 (300x20)	16	13977	937,5	930,99	0,699
133112 (300x10)	16	9897	1.723,99	1.694,2	1,758
133122 (300x20)	16	3124	1.603,9	1.392	15,223
133212 (300x10)	16	10501	1.242,6	1.198,3	3,697
133222 (300x20)	16	12164	1.163,9	1.151,8	1,051

4. Conclusões

Este trabalho trouxe uma heurística para o Problema de Localização e Roteamento em sua versão capacitada, que parte da estrutura do Recozimento Simulado com alguns incrementos. Assim, foram introduzidas funções de pertubação e diversificação para melhor explorar o espaço de soluções e tentar escapar de ótimos locais. Soma-se a isto um total de sete operadores para gerar

novas soluções, os quais são baseados em movimentos de troca e inserção de elementos do vetor solução.

Pode-se concluir que a heurística: (i) funcionou satisfatoriamente bem em termos de solução, com GAP médio de $1,896\%$ para as instâncias de Barreto (2004), $4,149\%$ para Prins et al. (2006) e $3,664\%$ para Tuzun e Burke (1999); e, (ii) precisou de um tempo computacional alto, em particular devido ao número total de iterações permitidas, uma vez que o parâmetro contador para o *NN* é reiniciado sempre que a melhor solução tem seu valor melhorado.

Um trabalho futuro que está em andamento consiste em analisar o motivo das soluções ruins. Já foi notado que pouca ou nenhuma vez o primeiro elemento do vetor solução sofre modificação. Assim, fazendo com que o depósito ali alocado, ainda na solução inicial, permaneça até a solução final. Outro ponto é melhorar o tamanho do intervalo dos parâmetros que o *irace* precisa considerar, na tentativa de diminuir o tempo computacional, pois tem sido observado que a melhor solução é encontrada logo nas primeiras iterações.

Agradecimentos.

Os autores gostariam de agradecer o apoio financeiro recebido das fundações de amparo à pesquisa CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) e FAPEG (Fundação de Amparo à Pesquisa do Estado de Goiás).

Referências

S. S. Barreto. *Análise e Modelização de Problemas de Localização-Distribuição*. tese de doutoramento, Gestão Indústrial, Universidade de Aveiro, Aveiro, Portugal, 2004.

C. Contardo, J.-F. Cordeau e B. Gendron. 2014a, An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. *INFORMS Journal on Computing*, 26: 88–102.

C. Contardo, J.-F. Cordeau e B. Gendron. 2014b, A grasp + ilp-based metaheuristic for the capacitated location-routing problem. *Journal of Heuristics*, 20:1–38.

M. Drexl e M. Schneider. 2015, A survey of variants and extensions of the location-routing problem. *European Journal of Operational Research*, 241:283–308.

J. W. Escobar, R. Linfati e P. Toth. 2013, A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. *Computers & Operations Research*, 40(1):70–79.

M. R. Garey e D. S. Johnson. *Computers and intractability: a guide to the theory of NP-completeness*. San Francisco: Freeman, 1979.

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

B. Jarboui, H. Derbel, S. Hanafi e N. Mladenović. 2013, Variable neighborhood search for location routing. *Computers & Operations Research*, 40:47–57.

G. Laporte e Y. Norbert. 1981, An exact algorithm for minimizing routing and operating costs in depot location. *European Journal of Operational Research*, 6:224–226.

M. Lopez-Ibanez, J. Dubois-Lacoste, T. Stutzle e Mauro Birattari. The irace package, iterated race for automatic algorithm configuration. Tr/iridia/2011-004, IRIDIA, Université libre de Bruxelles, Belgium, 2011.

C. C. Prins, C. Prodhon e R. W. Calvo. 2006, Solving the capacitated location-routing problem by a grasp complemented by a learning process and a path relinking. *4OR: A Quarterly Journal of Operations Research*, 4(3):221–238.

C. Prodhon e C. Prins. 2014, A survey of recent research on location-routing problems. *European Journal of Operational Research*, 238:1–17.

D. Tuzun e L. I. Burke. 1999, A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, 116:87–99.

V. F. Yu, S.-W. Lin, W. Lee e C.-J. Ting. 2010, A simulated annealing heuristic for the capacitated location routing problem. *Computers and Industrial Engineering*, 58:288–299.