Capítulo 6

Heurística baseada na vida de algas com aplicação no problema de localização e roteamento

Kamyla Maria Ferreira[1] Thiago Alves de Queiroz[1]

Resumo: Este trabalho trata da resolução do problema integrado de localização de instalações e roteamento de veículos capacitado, em que tanto os veículos quanto os depósitos são limitados em capacidade. A resolução do problema é feita pela aplicação do algoritmo artificial de algas, com o objetivo de minimizar o custo total relacionado com as decisões de localizar os depósitos e determinar rotas para entregar mercadorias aos clientes. A heurística desenvolvida é composta por três fases, quais sejam: movimento helicoidal, operação de reprodução e operação de adaptação. Experimentos computacionais foram realizados para testar o desempenho do algoritmo, em que as análises foram feitas a partir da comparação com resultados da literatura.1

Palavras-chave: Problema de Localização e Roteamento. Algoritmo Artificial de Algas. Heurística.

Introdução

O foco deste trabalho é estudar o problema integrado de localização e roteamento, uma vez que no ramo logístico torna-se importante tomar decisões que minimizem o custo total relacionado com a localização de depósitos e a determinação de rotas para a distribuição das mercadorias aos clientes. O Problema de Localização e Roteamento (PLR) é uma combinação de dois problemas de otimização, o Problema de Localização de Instalações e o Problema de Roteamento de

1 Universidade Federal de Goiás – UFG. Regional Catalão, Unidade Acadêmica Especial de Matemática e Tecnologia. Contato: kamylamaarria@gmail.com; taq@ufg.br.

Veículos, os quais são considerados NP-difíceis conforme Garey e Johnson (1979) e, assim, segue que o PLR também é NP-difícil.

O problema de localização de instalações consiste em definir quais instalações (depósitos, armazéns, fábricas, centros de distribuição, etc.) devem ser abertas para atender a demanda de clientes, de modo que o custo total de abertura seja mínimo. O problema de roteamento de veículos almeja traçar rotas de veículos para realizar a entrega de mercadorias aos clientes, com o intuito de minimizar o custo total de percorrer as rotas para atender toda a demanda dos clientes. A junção desses dois problemas resulta no PLR, que possui aplicações na área de distribuição de mercadorias (PERL; DASKIN, 1985), coleta de resíduos (KULCAR, 1996), projeto de rede óptica (LEE et al., 2003), distribuição de bens de consumo (AKSEN; ALTINKEMER, 2008), fornecimento de alimentos perecíveis (GOVINDAN et al., 2014), entre outros.

A resolução do PLR por algoritmos exatos, que retornam a solução ótima, foi realizada nos trabalhos de Laporte e Nobert (1981), que propuseram um algoritmo *branch-and-bound*, Belenguer et al. (2011), que desenvolveram um algoritmo do tipo *branch-and-cut*, Contardo, Cordeau e Gendron (2014a), que utilizaram um algoritmo *branch-and-cut-and-price,* e Ponboon, Qureshi e Taniguchi (2016), que desenvolveram um algoritmo *branch-and-price*.

As heurísticas, que embora não garantam a solução ótima, mas consomem baixo tempo computacional, para resolver o PLR, podem ser encontradas em Yu et al. (2010) e Zhang et al. (2014), que propuseram heurísticas baseadas no método de Recozimento Simulado, Contardo, Cordeau e Gendron (2014b), que desenvolveram uma heurística híbrida envolvendo o GRASP (*Greedy Randomized Adaptive Search Procedure*) combinada com a resolução de modelos de programação inteira, Marinakis (2015), que utilizou uma heurística de exame de partículas, e Koç et al. (2016), que desenvolveram um algoritmo híbrido de busca evolucionária.

Embora o uso de heurísticas seja adequado para aplicações reais, elas possuem parâmetros de entrada, os quais influenciam na convergência para boas soluções. A obtenção dos parâmetros pode ser feita manualmente ou com base na utilização de métodos de configuração automática. Um método desenvolvido por López-Ibánez et al. (2011), denominado *irace* (*Iterated Race for Automatic Algorithm Configuration*), encontra as configurações mais adequadas para os parâmetros a partir de um conjunto de instâncias, intervalos válidos para os parâmetros e da função a ser avaliada.

Vários estudos sobre PLR e suas variantes foram desenvolvidos recentemente. Conforme Prodhon e Prins (2014), entre os anos de 2007 e 2013 foram publicados cerca de 72 artigos sobre o PLR e suas diferentes versões. Diante disto, este trabalho busca contribuir com a literatura do PLR na sua variante capacitada, a

partir da aplicação do algoritmo artificial de algas, em que a calibração dos parâmetros é feita utilizando um método de configuração automática. Além disso, os resultados obtidos pela heurística devem ser comparados com outros da literatura a fim de mostrar a qualidade da heurística desenvolvida.

Este trabalho está organizado da seguinte forma: a próxima seção traz uma breve revisão da literatura sobre o PLR. A seção 2 caracteriza o PLR e mostra como o algoritmo artificial de algas foi adaptado para resolvê-lo. A seção 3 traz os experimentos computacionais sobre instâncias da literatura, sendo que conclusões e direções para trabalhos futuros são dados em seguida.

1 Estado-da-arte sobre o PLR

Laporte e Nobert (1981) abordaram o problema de localização e roteamento tratado por um algoritmo do tipo *branch-and-bound*. A formulação do problema foi através de um modelo linear inteiro, cuja versão consistia em definir uma única instalação sujeita a um número de veículos fixo. Além disso, o comprimento das rotas não possuía restrições. A validação do método foi dada por intermédio de um conjunto de instâncias randômicos variando de 20 a 50 cidades.

Tuzun e Burke (1999) estudaram a versão do PLR cujas rotas são capacitadas e os depósitos incapacitados, isto é, as demandas dos clientes em uma rota não podem ultrapassar a capacidade do veículo, enquanto que o depósito pode atender demandas ilimitadas. O problema foi resolvido por meio da abordagem de busca Tabu em duas fases sobre vários níveis de decisão. Os testes computacionais foram realizados em 36 instâncias, as quais são compostas por 100, 150 ou 200 clientes, com 10 ou 20 locais de depósitos.

Albareda-Sambola, Díaz e Fernández (2005) desenvolveram para o PLR um algoritmo de busca Tabu com rotinas para o aperfeiçoamento das rotas e realizar a diversificação com relação aos depósitos abertos. Os autores também resolveram uma formulação compacta para encontrar um conjunto de caminhos em uma rede auxiliar. Para realizar a validação do algoritmo, os autores geraram algumas instâncias aleatoriamente, contendo no máximo 30 clientes.

Escobar e Linfati (2012) resolveram o PLR capacitado por meio de uma heurística baseada no recozimento simulado e na busca granular. Os experimentos computacionais foram realizados em dois conjuntos de instâncias da literatura, sendo compostos por clientes variando no intervalo de 20 a 200 e os depósitos de 5 a 20. Os resultados obtidos mostraram que a heurística desenvolvida é eficiente, com soluções de qualidade sendo obtidas dentro de um tempo computacional aceitável para aplicações práticas.

Jarboui et al. (2013) desenvolveram uma heurística inspirada na busca em vizinhança variável para resolver o PLR. Na versão do problema estudado, cada

veículo é responsável por realizar apenas uma rota estando associado somente a um depósito. A validação do algoritmo foi efetuada experimentalmente por meio de um conjunto de 79 instâncias, com o número de clientes variando de 20 a 200 e as instalações de 5 a 20.

Mais recentemente, Koç et al. (2016) desenvolveram um algoritmo híbrido de busca evolucionária para resolver o PLR com frota heterogênea e janela de tempo. Os experimentos computacionais foram realizados em um conjunto de 56 instâncias com 100 clientes e 10 depósitos, mostrando que o algoritmo é eficiente na resolução do problema.

Lopes, Ferreira e Santos (2016) propuseram um algoritmo genético híbrido para resolver a versão capacitada do PLR. Para comprovar a eficiência da heurística, testes computacionais foram realizados sobre 85 instâncias da literatura, as quais são compostas por clientes variando de 20 a 200 e de 5 a 20 depósitos. Os resultados mostraram que a heurística desenvolvida é capaz de resolver problemas práticos em um tempo computacional baixo.

Ponboon, Qureshi e Taniguchi (2016) desenvolveram um algoritmo exato do tipo *branch-and-price* para resolver o PLR com janela de tempo. Segundo os autores, ainda não havia trabalhos relacionados com a aplicação de algoritmos exatos para a resolução desta versão do PLR. A validação do algoritmo foi feita por intermédio de testes computacionais em três grupos de instâncias (com depósitos de capacidade pequena, média e grande) mostrando que o algoritmo proposto é eficiente na resolução do problema.

2 Caracterização do problema e da heurística

Esta seção traz a definição do problema integrado de localização de instalações e roteamento de veículos. Em seguida, apresenta-se uma descrição do algoritmo artificial de algas desenvolvido.

2.1 Descrição do PLR

Conforme Prins et al. (2007), a variante capacitada do PLR pode ser definida em um grafo não-orientado $G = (V, E)$, tal que V representa o conjunto de vértices (ou nós), composto pelo subconjunto de depósitos $I = \{1, 2, \ldots, m\}$ e pelo subconjunto de clientes $J = \{1 + m, 2 + m, \ldots, n + m\}$, e E representa todas as arestas que ligam dois nós de V, com exceção de arestas que conectam depósito com depósito. Cada aresta $(i, j) \in E$ tem uma distância/custo dada por $c_{ij} > 0$. Além disso, cada cliente $j \in J$ tem demanda d_j que deve ser atendida por um único veículo e uma única vez. Cada depósito $i \in I$ tem capacidade máxima de distribuição W_i e custo de instalação O_i.

Existe um conjunto de K veículos idênticos, cada um com capacidade máxima de carga Q. Para cada veículo utilizado por um depósito, incorre um custo fixo F, sendo que um veículo só pode executar uma única rota. Cada rota começa e termina no mesmo depósito, tal que a capacidade Q do veículo não pode ser excedida. Também, um depósito pode ter mais de uma rota, todavia a sua capacidade de distribuição deve ser respeitada. O objetivo é minimizar o custo total incorrido da instalação de depósitos, das rotas partindo dos depósitos e do uso dos veículos, tal que as restrições mencionadas devem ser atendidas.

2.2 Descrição do algoritmo artificial de algas

O Algoritmo Artificial de Algas (*AAA - Artificial Algae Algorithm*) para o PLR capacitado, denominado por AAAPLR, desenvolvido neste trabalho, segue a estrutura do algoritmo proposto por Uymaz, Tezel e Yel (2015). Algumas adaptações foram realizadas, uma vez que o AAA foi apresentado para problemas contínuos de maximização.

O Algoritmo Artificial de Algas (AAA) simula a vida das algas que consiste: em seu modo de mover/nadar, dado por um movimento helicoidal; adaptação a novos ambientes; e, a reprodução por divisão mitótica. Assim, o algoritmo é composto por três fases principais: movimento helicoidal, adaptação e reprodução. O Algoritmo 1 apresenta a estrutura completa do AAAPLR, que possui 7 parâmetros de entrada, quais sejam:

- P_a: probabilidade de aplicar a operação de adaptação;
- I_{iter}: número máximo de iterações;
- N_n: quantidade de colônias de algas;
- e: taxa de perda de energia;
- P: penalidade aplicada quando ocorre violação da capacidade do depósito;
- f_c: força de cisalhamento; e,
- ε: tolerância.

No início do Algoritmo 1, gera-se uma população inicial z, dada por uma matriz de ordem $N_n \times L$, em que cada linha representa um grupo de algas que vivem juntas, denominado de colônia. A quantidade de algas em uma colônia é dada na eq. (2.1).

$$L = N + M + N_{zero}, \tag{2.1}$$

em que N é a quantidade de cliente, M é o numero de depósitos e N_{zero} corresponde a quantidade mínima de veículos (ou rotas) necessária para tender a demanda de todos os clientes. O N_{zero} é utilizado para terminar as rotas, auxiliando na obtenção de soluções melhores, uma vez sem os zeros as rotas só seriam

terminadas quando a capacidade do veículo fosse atingida. Porém, ao considerar os zeros as rotas podem ser rescindidas com qualquer quantidade de demanda inferior a capacidade de carga Q do veículo. O cálculo do N_{zero} é dado por:

$$N_{zero} = \left\lceil \frac{\sum_{j=1}^{N} d_j}{Q} \right\rceil \tag{2.2}$$

Algoritmo 1 – Estrutura Geral do AAAPLR.

Entrada: P_a, I_{iter}, N_n, e, P, f_c, ε

Gerar a população z com as colônias de algas iniciais;
$T_t \leftarrow 1, \quad t = 1,2,\ldots,N_n;$
$Ap_t \leftarrow 0, \quad t = 1,2,\ldots,N_n;$
$I \leftarrow 0, \quad Fbest \leftarrow min(F(z,P));$
Enquanto $I \neq I_{iter}$ **faça**
  $E_t \leftarrow \left\lceil \frac{N_n}{T_t} \right\rceil, \quad w_t \leftarrow 2\pi \left(\sqrt[s]{\frac{3T_t}{4\pi}} \right)^2, \quad t = 1,2,\ldots,N_n;$
  para $t = 1$ *até* N_n **faça**
    $nf \leftarrow True;$
    Enquanto $E_t \geqslant 0$ **faça**
      $Y \leftarrow$ Aplicar o **Movimento Helicoidal** em $z_t;$
      $E_t \leftarrow E_t - e/2;$
      Se $F(Y,P) < F(z_t,P)$ **então**
        $F(z_t,P) \leftarrow F(Y,P); \quad z_t \leftarrow Y; \quad nf \leftarrow False;$
      Senão $E_t \leftarrow E_t - e/2;$
    Se $nf = True$ **então** $Ap_t \leftarrow Ap_t + 1;$
  $T_t \leftarrow T_t + T_t \frac{F(z_t P)}{T_t + F(z_t P)}, \quad t = 1,2,\ldots,N_n;$
  Aplicar a **Operação de Reprodução** em $z;$
  Gerar um número aleatório $r;$
  Se $r < P_a$ **então** Aplicar a **Operação de Adaptação** em $z;$
  $I \leftarrow I + 1;$
  Se $erro < \varepsilon$ **então** $I \leftarrow I_{iter} + 1;$
Saída: Colônia z_t com menor valor de $F(z_t,P)$.

Para gerar a solução inicial, usa-se um procedimento aleatório descrito na subseção 2.2.2. Posteriormente, atribuem-se os valores iniciais do tamanho da população (T_t) e a aptidão, sendo que Ap_t corresponde a aptidão de uma colônia

de alga e está relacionado com a melhora ou piora do valor da função objetivo. O tamanho da população representa as condições de vida das algas, tal que quando as algas estão em condições de vida perfeita, elas vão reproduzir e aumentar de tamanho.

Em seguida, calcula-se a energia da população fazendo o número de algas na colônia sobre o seu tamanho, assim quanto menor o tamanho da colônia tt, maior deve ser a sua energia. O próximo passo é a aplicação do movimento helicoidal, o qual busca movimentar as colônias em direção a colônia com melhor valor de função objetivo. O movimento helicoidal ocorre em três dimensões para cada colônia. Dessa forma, três algas da colônia são selecionadas de forma aleatória e as suas posições são modificadas conforme as eqs. (2.3) a (2.5), respectivamente,

$$z_{tv} = z_{tv} + (z_{qv} - z_{tv})(f_c - w_t)\rho, \tag{2.3}$$

$$z_{tw} = z_{tw} + (z_{qw} - z_{tw})(f_c - w_t)cos(\alpha), \tag{2.4}$$

$$z_{tl} = z_{tl} + (z_{ql} - z_{tl})(f_c - w_t)sin(\beta), \tag{2.5}$$

sendo que v, w e l são números inteiros escolhidos aleatoriamente entre o intervalo $[1, L]$; q corresponde a uma colônia vizinha escolhida de forma aleatória entre $[1, N]$; ρ é um número real que varia no intervalo $[-1, 1]$; e, por fim, α e β estão no intervalo de $[0, 2\pi]$.

Após a geração da nova solução Y, realiza-se a avaliação da mesma, assim a solução Y é aceita se $F(Y,P) < F(z_t,P)$. Quando a energia de todas as colônias acaba, atualiza-se o tamanho T_t da colônia t e, então, a solução z é submetida a operação de reprodução. Quando as algas recebem nutrientes e luz em quantidades adequadas, elas crescem e se reproduzem rapidamente, caso contrário, com baixos índices de nutrientes e luz, as algas sobrevivem até um certo período de tempo e, posteriormente, morrem. Assim, no AAA, a colônia z_t fica maior ao se mover em direção a solução mais eficiente, ou seja, com melhor valor de função objetivo, sendo este processo apresentado nas eqs. (2.6) a (2.8),

$$alga\_maior = \max_{t=1,2,\ldots,N_n} (T_t), \tag{2.6}$$

$$alga\_menor = \min_{t=1,2,\ldots,N_n} (T_t), \tag{2.7}$$

$$alga\_maior_i = alga\_menor_i, \qquad i = 1, 2, \dots, L. \tag{2.8}$$

Em seguida, no caso de um número aleatório r ser menor do que P_a, aplica-se a operação de adaptação, na qual as colônias que não podem crescer o suficiente tentam se adaptar ao ambiente. A adaptação é o processo em que uma colônia busca se assemelhar a melhor colônia da população. O movimento de adaptação consiste em escolher a colônia d com maior Ap e adaptá-la conforme as eqs. (2.9) e (2.10),

$$z\_d = \max_{t=1,2,\dots,N_n} (Ap_t), \tag{2.9}$$

$$z_{di} = z_{di} + (alga\_menor_i - z_{di})r_2, \ \ se \ \ r < P_a, \quad i = 1,2, \dots, L. \tag{2.10}$$

O algoritmo chega ao fim quando o erro relativo ($erro$) for menor do que um parâmetro ϵ ou quando I_{iter} iterações são executadas. O erro relativo é dado por:

$$erro = \sqrt{\sum \left(F(zant_t, P) - F(z_t, P)\right)^2}. \tag{2.11}$$

2.2.1 Função de avaliação

Uma solução Y é avaliada pela função $F(Y,P)$ a ser minimizada, sendo que seu cálculo é dado por:

$$\sum_{i \in I} O_i y_i + \sum_{k \in K} \sum_{(i,j) \in E} c_{ij} x_{ijk} + \sum_{k \in K} \sum_{i \in I} \sum_{j \in J} F x_{ijk} + P \sum_{i \in I} A(i), \tag{2.12}$$

em que a primeira parcela desta função corresponde ao custo de abertura do depósito, enquanto que a segunda está associada ao custo das rotas, que consiste na distância percorrida para cada rota. A terceira parcela está relacionada com o custo fixo de utilizar os veículos e, por fim, há uma última parcela associada a penalidade aplicada ao excesso de demanda do depósito, isto é, toda vez que a capacidade do depósito for ultrapassada, o valor excedido é multiplicado por um fator P.

A ideia é que a heurística convirja, ao passar das iterações, para soluções que respeitem a capacidade dos depósitos abertos, visto que pagar a penalidade torna a solução não atrativa. Para realizar o cálculo da penalidade, seja $D(i)$ a diferença entre a soma da demanda dos clientes j atendidos pelo depósito i pela capacidade do depósito i. Logo, $A(i) = max\left(0, D(i)\right)^2$.

2.2.2 População inicial

A população inicial é gerada de forma aleatória conforme o Algoritmo 2, que inicialmente gera uma matriz com a sequência de números de 1 até $N + M$, além de considerar os N_{zero}. Em seguida, cada linha da matriz passa por um embaralhamento ($shuffle$) de forma aleatória, o que fornece uma solução para o PLR.

Algoritmo 2 – Rotina para gerar a solução inicial.

1 **para** $i = 0$ *até* N_n **faça**
2 **para** $j = 0$ *até* $N + M$ **faça**
3 $z[i][j] = j + 1;$
4 **para** $i = 0$ *até* N_n **faça**
5 $shuffle(z[i]);$

Saída: População z.

3 Experimentos Computacionais

A heurística AAAPLR foi codificada em linguagem *Python*. Os testes computacionais foram realizados em um computador com processador Intel Core i7-4790K de 4 GHz, 32 GB de memória RAM e sistema operacional Linux Ubuntu 14.04 LTS. A solução de cada instância foi obtida a partir da realização de uma única execução da heurística.

Os parâmetros iniciais da heurística foram calibrados pelo pacote *irace*, em que somente o valor de ε e P não foram calibrados. O ε foi tomado fixo em 10^{-10}, uma vez que ele está relacionado com um dos critérios de parada, tal que um valor alto pode resultar em convergência prematura. O valor de P foi determinado para ser $200(\sum O_i / \sum W_i)$. Após a calibração dos parâmetros, obteve-se: $I_{iter} = 4941,\ N_n = 88,\ f_c = 9, e = 0{,}16,\ P_a = 0{,}47$ e semente igual a *491*.

A validação da heurística ocorreu de forma numérica com experimentos sobre 27 instâncias da literatura. As instâncias escolhidas fazem parte do conjunto proposto por Barreto (2004) e Prins, Prodhon e Calvo (2006). As instâncias possuem as informações do número de clientes e de depósitos disponíveis, coordenadas para os clientes, coordenadas para os depósitos, capacidade do veículo, capacidade do depósito, demanda dos clientes, custo de abertura para os depósitos e custo de utilização de um veículo.

As Tabelas 3.1 e 3.2 apresentam os resultados obtidos para as instâncias de Barreto (2004) e Prins, Prodhon e Calvo (2006), respectivamente. As tabelas contêm as informações: quantidade de clientes *n* e depósitos *m*; valor da melhor

solução conhecida (*BKS - Best Known Solution*), obtida a partir dos trabalhos dos autores apresentados na própria tabela; soluções encontradas por outros três métodos: SALRP de Yu et al. (2010), MACO de Ting e Chen (2013) e GRASP+ILP de Contardo, Cordeau e Gendron (2014b); valor da solução encontrado pela heurística AAAPLR e o respectivo tempo gasto em segundos; o número de avaliações da função objetivo; e, a diferença relativa (GAP em porcentagem) entre o valor da solução encontrada e a melhor solução conhecida.

A Tabela 3.1 sumariza os resultados obtidos para as instâncias de Barreto (2004). Ao analisar o GAP, percebe-se que o AAAPLR não apresentou um desempenho satisfatório, uma vez que a diferença relativa, por exemplo, da instância Gaspelle5 (32x5) foi maior do que 21%. O tempo computacional médio foi de 611,47 segundos e o GAP médio foi de 11,06%, sendo superior aos resultados dos demais métodos da literatura.

Tabela 6.1 Resultados para as instâncias de Barreto (2004).

Instância (nxm)	BKS	SALRP		GRASP+ILP		MACO		AAALPR			
		Solução	GAP (%)	Solução	GAP (%)	Solução	GAP (%)	Solução	Avaliações da Função	CPU	GAP (%)
Gaspelle (21x5)	424,90	424,90	0,00	424,90	0,00	424,90	0,00	441,27	160549	456,12	3,85
Gaspelle2 (22x5)	585,10	585,10	0,00	585,10	0,00	585,10	0,00	591,04	172319	487,67	1,02
Gaspelle3 (29x5)	512,10	512,10	0,00	512,10	0,00	512,10	0,00	604,47	170505	580,45	18,04
Gaspelle4 (32x5)	562,20	562,20	0,00	562,22	0,00	562,22	0,00	633,10	191420	715,74	12,61
Gaspelle5 (32x5)	504,30	504,30	0,00	504,30	0,00	504,30	0,00	611,82	178411	653,59	21,32
Gaspelle6 (36x5)	460,40	460,40	0,00	460,40	0,00	460,40	0,00	495,23	179731	717,38	7,56
Min27 (27x5)	3062,00	3062,00	0,00	3062,00	0,00	3062,00	0,00	3460,99	200394	669,34	13,03
Média Global			**0,00**		**0,00**		**0,00**			**611,47**	**11,06**

Na Tabela 3.2 é possível observar que a solução BKS não foi encontrada para nenhuma das instâncias. As instâncias com 20 clientes e 5 depósitos apresentaram o menor GAP comparada com as demais instâncias, contudo a heurística desenvolvida não consegui obter nenhum resultado melhor do que os algoritmos SALRP, MACO e GRASP+ILP. O tempo de execução médio para estas instâncias foi de 346,86 segundos. Para as instâncias com 5 depósitos e 50 clientes, obteve-se uma diferença relativa média de 20,86%, com tempo computacional médio de 1.615,61 segundos.

Para as instâncias com 100 clientes e 5 depósitos, encontrou-se uma solução com uma diferença relativa de 35,32%, sendo a diferença média dessas instâncias igual a 25,28%, um valor superior ao dos outros métodos da literatura. Por fim, tem-se as instâncias com 100 clientes e 10 depósitos, cujos resultados tiveram um GAP médio superior a 20%. O GAP médio considerando todas as instâncias foi de 18,16%, enquanto o tempo computacional médio foi de 2.666,49 segundos.

Tabela 6.2 Resultados para as instâncias de Prins, Prodhon e Calvo (2006).

Instância (nxm)	BKS	SALRP		GRASP+ILP		MACO		AAALPR			
		Solução	GAP (%)	Solução	GAP (%)	Solução	GAP (%)	Solução	Avaliações da Função	CPU	GAP (%)
20-5-1 (20x5)	54793	54793	0,00	54793	0,00	54793	0,00	55021	169277	146,74	0,42
20-5-1b (20x5)	39104	39104	0,00	39104	0,00	39104	0,00	39492	183541	362,53	0,99
20-5-2 (20x5)	48908	48908	0,00	48908	0,00	48908	0,00	49199	165139	463,49	0,59
20-5-2b (20x5)	37542	37542	0,00	37542	0,00	37542	0,00	37611	155285	414,69	0,18
Média			**0,00**		**0,00**		**0,00**		**306789**	**346,86**	**0,55**
50-5-1 (50x5)	90111	90111	0,00	90111	0,00	90111	0,00	98752		1615,65	9,59
50-5-1b (50x5)	63242	63242	0,00	63242	0,00	63242	0,00	82428	299272	1620,41	30,34
50-5-2 (50x5)	88298	88298	0,00	88298	0,00	88298	0,00	100594	259717	1510,81	13,93
50-5-2b (50x5)	67308	67308	0,00	67373	0,10	67308	0,00	78187	362308	1938,02	16,16

Continua

Tabela 6.2 Resultados para as instâncias de Prins, Prodhon e Calvo (2006). *(Continuação)*

Instância (nxm)	BKS	SALRP		GRASP+ILP		MACO		AAALPR			
		Solução	GAP (%)	Solução	GAP (%)	Solução	GAP (%)	Solução	Avaliações da Função	CPU	GAP (%)
50-5-2bBIS (50x5)	84055	84055	0,00	84055	0,00	84055	0,00	93411	314884	1639,55	11,13
50-5-2BIS (50x5)	51822	51822	0,00	51883	0,12	51822	0,00	67950	284508	1622,94	31,12
50-5-3 (50x5)	86203	86203	0,00	86203	0,00	86203	0,00	98561	267430	1539,59	14,34
50-5-3b (50x5)	61830	61830	0,00	61830	0,00	61830	0,00	75087	267545	1437,94	21,44
Média			**0,00**		**0,03**		**0,00**		**556626**	**1615,61**	**18,51**
100-5-1 (100x5)	275419	275419	0,00	275457	0,01	276220	0,29	324134		6042,23	17,69
100-5-1b (100x5)	213615	213615	0,00	214056	0,21	214323	0,33	269311	508285	4462,82	26,07
100-5-2 (100x5)	193671	193671	0,00	193708	0,02	194441	0,40	226837	430290	4301,37	17,12
100-5-2b (100x5)	157095	157150	0,04	157178	0,05	157222	0,08	204900	655479	5428,07	30,43

Continua

Tabela 6.2 Resultados para as instâncias de Prins, Prodhon e Calvo (2006). *(Continuação)*

Instância (nxm)	BKS	SALRP		GRASP+ILP		MACO		AAALPR			
		Solução	GAP (%)	Solução	GAP (%)	Solução	GAP (%)	Solução	Avaliações da Função	CPU	GAP (%)
100-5-3 (100x5)	200079	200079	0,00	200339	0,13	201038	0,48	250141	502081	5352,65	25,02
100-5-3b (100x5)	152441	152441	0,00	152466	0,02	152722	0,18	206286	471314	3995,80	35,32
Média			**0,01**		**0,07**		**0,29**			**4930,49**	**25,28**
100-10-1 (100x10)	287892	287983	0,03	287892	0,00	291134	1,13	357387	422355	4039,64	24,14
100-10-1b (100x10)	231763	231763	0,00	234080	1,00	235348	1,55	318044	568922	5394,80	37,23
Média			**0,02**		**0,50**		**1,34**			**4717,22**	**30,68**
Média Global			**0,00**		**0,08**		**0,22**			**2666,49**	**18,16**

Conclusões e trabalhos futuros

Este trabalho apresentou o algoritmo artificial de algas para resolver o problema integrado de localização e roteamento. Considerou-se a versão capacitada do problema, em que os depósitos e os veículos têm capacidade limitada. A heurística desenvolvida parte da estrutura proposta por Uymaz, Tezel e Yel (2015), com a proposta de adaptações nos parâmetros e nas operações de reprodução e adaptação de forma a ser aplicada no PLR.

A heurística foi testada em 27 instâncias de dois autores diferentes, tal que para pequenas instâncias ela apresentou um desempenho satisfatório. Por outro lado, a heurística teve resultados ruins para instâncias grandes, tendo um desempenho geral além do esperado. Assim, para as instâncias de Prins, Prodhon e Calvo (2006), o GAP médio foi de 18,16% e o tempo computacional médio foi de 2.666,49 segundos, sendo as maiores instâncias compostas por 100 clientes e 10 depósitos. Para as sete instâncias avaliadas de Barreto (2004), o GAP médio foi de 11,06% com um tempo computacional médio de 611,47 segundos.

Trabalhos futuros desta pesquisa visam estudar os parâmetros iniciais da heurística, em especial o *e* e a fórmula para calcular a energia das colônias, uma vez que a energia impacta no número de aplicações da função helicoidal. Além disso, objetiva-se desenvolver alguns operadores para ajudar a explorar melhor o espaço de busca, visto que ao longo dos experimentos, notou-se que a heurística fica estagnada em possíveis ótimos locais.

Agradecimentos

Os autores agradecem o apoio financeiro recebido do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e da Fundação de Amparo à Pesquisa do Estado de Goiás (FAPEG).

Referências

AKSEN, D.; ALTINKEMER, K. A location-routing problem for the conversion to the "click-and-mortar" retailing: The static case. *European Journal of Operational Research*, v. 186, n. 2, p. 554-575, 2008.

ALBAREDA-SAMBOLA, M.; DÍAZ, J. A.; FERNÁNDEZ, E. A compact model and tight bounds for a combined location-routing problem. *Computers & Operations Research*, v. 32, n. 3, p. 407–428, 2005.

BARRETO, S. S. *Análise e Modelização de Problemas de Localização-Distribuição*. Tese (Doutorado) – Gestão Industrial, Universidade de Aveiro, Aveiro, Portugal, 2004.

BELENGUER, J.-M. et al. A branch-and-cut method for the capacitated location-routing problem. *Computers & Operations Research*, v. 38, n. 6, p. 931-941, 2011.

CONTARDO, C.; CORDEAU, J.-F.; GENDRON, B. An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. *INFORMS Journal on Computing*, v. 26, n. 1, p. 88-102, 2014a.

CONTARDO, C.; CORDEAU, J.-F.; GENDRON, B. A GRASP + ILP-based metaheuristic for the capacitated location-routing problem. *Journal of Heuristics*, v. 20, n. 1, p. 1-38, 2014b.

ESCOBAR, J. W.; LINFATI, R. Un algoritmo metaheurístico basado en recocido simulado con espacio de búsqueda granular para el problema de localización y ruteo con restricciones de capacidad. *Revista Ingenierías Universidad de Medellín*, v. 11, n. 21, p. 139-150, 2012.

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

GOVINDAN, K. et al. Two-echelon multiple-vehicle location–routing problem with time windows for optimization of sustainable supply chain network of perishable food. *International Journal of Production Economics*, v. 152, n. 1, p. 9-8, 2014.

JARBOUI, B. et al. Variable neighborhood search for location routing. *Computers & Operations Research*, v. 40, n. 1, p. 47-57, 2013.

KOÇ, Ç. et al. The fleet size and mix location-routing problem with time windows: Formulations and a heuristic algorithm. *European Journal of Operational Research*, v. 248, n. 1, p. 33–51, 2016.

KULCAR, T. Optimizing solid waste collection in brussels. *European Journal of Operational Research*, v. 90, n. 1, p. 71–77, 1996.

LAPORTE, G.; NOBERT, Y. An exact algorithm for minimizing routing and operating costs in depot location. *European Journal of Operational Research*, v. 6, n. 2, p. 224-226, 1981.

LEE, Y. et al. A location-routing problem in designing optical internet access with wdm systems. *Photonic Network Communications*, v. 6, n. 2, p. 151-160, 2003.

LÓPEZ-IBÁNEZ, M. et al. *The irace package, Iterated Race for Automatic Algorithm Configuration*. Relatório Técnico, 2011.

LOPES, R. B.; FERREIRA, C.; SANTOS, B. S. A simple and effective evolutionary algorithm for the capacitated location-routing problem. *Computers & Operations Research*, v. 70, p. 155-162, 2016.

MARINAKIS, Y. An improved particle swarm optimization algorithm for the capacitated location routing problem and for the location routing problem with stochastic demands. *Applied Soft Computing*, v. 37, p. 680-701, 2015.

PERL, J.; DASKIN, M. S. A warehouse location-routing problem. *Transportation Research Part B: Methodological*, v. 19, n. 5, p. 381-396, 1985.

PONBOON, S.; QURESHI, A. G.; TANIGUCHI, E. Branch-and-price algorithm for the location-routing problem with time windows. *Transportation Research Part E: Logistics and Transportation Review*, v. 86, p. 1-19, 2016.

PRINS, C. et al. Solving the capacitated location-routing problem by a cooperative lagrangean relaxation-granular tabu search heuristic. *Transportation Science*, v. 41, n. 4, p. 470-483, 2007.

PRINS, C. C.; PRODHON, C.; CALVO, R. W. 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*, v. 4, n. 3, p. 221-238, 2006.

PRODHON, C.; PRINS, C. A survey of recent research on location-routing problems. *European Journal of Operational Research*, v. 238, p. 1-17, 2014.

TING, C.-J.; CHEN, C.-H. A multiple ant colony optimization algorithm for the capacitated location routing problem. *International Journal of Production Economics*, v. 141, n. 1, p. 34-44, 2013.

TUZUN, D.; BURKE, L. I. A two-phase tabu search approach to the location routing problem. *European Journal of Operational Research*, v. 116, p. 87-99, 1999.

UYMAZ, S. A.; TEZEL, G.; YEL, E. Artificial Algae Algorithm (AAA) for nonlinear global optimization. *Applied Soft Computing*, v. 31, p. 153-171, 2015.

YU, V. F. et al. A simulated annealing heuristic for the capacitated location routing problem. *Computers & Industrial Engineering*, v. 58, p. 288-299, 2010.

ZHANG, Y. et al. Hybrid metaheuristic solutions to inventory location routing problem. *Transportation Research Part E: Logistics and Transportation Review,* v. 70, p. 305-323, 2014.