doi:10.20396/revpibic262018343

Algoritmo Evolutivo para o Problema de Distritamento-Roteamento

Davi Rodrigues*, Fabio L. Usberti, Celso Cavellucci.

Resumo
O problema do caixeiro-viajante (TSP) propõe encontrar o menor caminho possível para se percorrer um grafo G = (V, E), visitando todos os nós V através de suas arestas E, das quais a cada uma é atribuído um custo. Uma generalização do TSP é o problema de distritamento-roteamento, onde o objetivo é particionar o grafo em um número pré-determinado de distritos conexos, onde cada distrito será percorrido por um veículo próprio, de forma a se obter o menor caminho total. O objetivo deste trabalho é propor uma metodologia de solução para o problema distritamento-roteamento através de uma meta-heurística baseada em algoritmos evolutivos.

Palavras-chave:
Meta-heurística, Algoritmos Genéticos, Otimização Combinatória.

Introdução

O problema de distritamento e roteamento (em inglês districtiong and routing problem - DRP) é uma generalização do problema do caixeiro-viajante (em inglês traveling salesman problem - TSP). O TSP consiste na busca de um ciclo hamiltoniano com o menor custo possível. O espaço de busca aumenta fatorialmente em relação ao número de nós, tornando a busca por uma solução exata impraticável para instâncias suficientemente grandes.
No DRP buscamos dividir o grafo em um número pré-determinado de distritos conexos, onde cada distrito será percorrido por um "caixeiro" (veículo) próprio. O DRP pode ser modelado como um grafo não-direcionado G = (V, E), onde V é a união do conjunto de clientes com o conjunto de depósitos; e E é o conjunto de arestas. Dependendo da aplicação, diferentes objetivos podem ser considerados para a otimização, como, por exemplo, número de distritos, somatório dos custos dos trajetos dos veículos, equilíbrio de trabalho entre os veículos, entre outros.
Este trabalho tem por objetivo propor, implementar e analisar metodologias heuristicas baseadas em algoritmos genéticos para a solução do DRP.

Resultados e Discussão

Os grafos foram modelados como uma matriz de adjacência de custos, e os cromossomos através de um vetor com n+k elementos, indicando a ordem em que os n nós são percorridos, utilizando k separadores para marcar o início de um novo ciclo. O algoritmo evolutivo utilizado segue o modelo proposto por Reeves, C.[1], utilizando o inverso do custo global como fitness, o método do torneio para seleção da população e recombinação simples para o crossover.
Para a criação da população inicial, foram comparados os métodos de geração aleatória, através de quadrados latinos[2], e através de um algoritmo guloso [3]. Ainda foi implementado o método de otimização 2-Opt[4], comparando o seu desempenho quando aplicado nas gerações intermediarias ou apenas na solução final.
As instâncias utilizadas foram a a280, ali535 e berlin52, da biblioteca TSPLib[5]. Segue os ensaios realizados para a instância a280, com seus respectivos custos globais:

Tabela 1. Resultados dos ensaios para a instância a280.

Ensaio	Custo
População gerada aleatoriamente.	2691
População gerada aleatoriamente, realizando busca local 2-opt na solução final.	2784
População gerada aleatoriamente, realizando 2-opt em cada cromossomo filho que entrará na próxima geração.	2794
População gerada através de algoritmo guloso.	3162
População gerada através de algoritmo guloso, realizando busca local 2-opt na solução final.	3527
População gerada através de algoritmo guloso, realizando 2-opt em cada cromossomo filho que entrará na próxima geração.	18953

Conclusão

Podemos notar que não houve mudanças expressivas de desempenho entre os ensaios 1, 2 e 3. Entretanto, como o 2opt possui complexidade quadrática[4], utilizá-lo várias vezes durante a execução deixou o algoritmo custoso, com cada geração levando cerca de 6 minutos para ser processada. Sendo assim apenas duas ou três gerações foram computadas durante o tempo de execução, o que justifica o desempenho ligeiramente inferior do ensaio 2 em relação ao ensaio 1. Tanto o algoritmo guloso quanto o 2opt acrescentaram em desempenho.

1 Reeves, C. *Genetic Algorithms for the Operations Researcher. Journal on Computing Vol. 9, No.3, Summer 1997, 231 – 250.*

2 *A. D. Keedwell; J. Denes. Latin Squares and Their Applications, 2a edição, North Holland, 2015.*

3 *Brassard, Gilles; Bratley, Paul. Algorithmics theory and practice. New Jersey. Prentice-Hall, 1988.*

4 *Johnson, David S.; McGeoch, Lyle A. Local Search in Combinatorial Optimization. E. H. L. Aarts e J. K. Lenstra (eds.), Londres, 1997. p. 215-318*

5 *Reinelt, G., Universität Heidelberg. Disponivel em: <http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html>, Acesso em 08 de jun. 2017.*