

Implementação de uma metaheurística para o Problema do Caixeiro Viajante

Ronaldo de Figueiredo Silveira

9 de junho de 2015

Resumo

O presente trabalho tem como objetivo a apresentação de um algoritmo meta-heurístico, utilizando a busca tabu, para o Problema do Caixeiro Viajante (PCV) bem como os resultados computacionais dos testes com as instâncias da literatura.

Palavras-Chave: PCV, Caixeiro Viajante, Algoritmo meta-heurístico, busca tabu

1 Introdução

O Problema do Caixeiro Viajante (PCV) não possui data definida, mas estima-se que no século XIX já se falava dele, apesar de ter sido realmente estudado no século XX em Harvard e Viena^[1]. Entretanto o problema, com esse nome, ficou mundialmente conhecido em 1950^[2]. PCV é um problema que visa encontrar o menor caminho com certas características, num conjunto de cidades e estradas que ligam essas cidades. As características são:

1. Deve passar por todas as cidades exatamente uma vez. Nem mais, nem menos.
2. Deve começar de uma cidade, digamos, v_0 e voltar à mesma cidade no final.

Assim, temos que o problema pode ser traduzido para: encontrar o menor ciclo hamiltoniano em um grafo.

O Problema do Caixeiro Viajante pode parecer bastante simples à primeira vista, já que é um problema que se assemelha a muitos problemas do mundo real. Entretanto esse, ao ser implementado, percebe-se a complexidade enorme de encontrar tal ciclo.

Esse problema pode ser definido, formalmente como: Dado um grafo $G = (V, E)$, onde V é o conjunto de vértices e E o conjunto de arestas, encontrar a permutação de vértices que forme um circuito hamiltoniano e minimize seu custo.

Em 1972, Richard Karp demonstrou que o problema do ciclo hamiltoniano é da classe NP-Completo^[3]. Sendo assim, seu equivalente em otimização, o Caixeiro Viajante, é um problema NP-Difícil.

2 A meta-heurística

2.1 Meta-heurísticas e busca tabu

Uma meta-heurística, é uma forma heurística de resolução de problemas genéricos de otimização^[4]. Isto é, a meta-heurística é uma espécie de framework que pode ser utilizado para resolver diversos problemas considerados difíceis (problemas NP-árduos como o PCV supracitados são exemplos).

A meta-heurística se diferencia da heurística no que se diz respeito a utilização. A primeira é utilizada de modo geral, em diversos tipos de problemas de otimização, enquanto a segunda é específica para um problema. Como exemplos temos a meta-heurística Simulated Annealing e a heurística da inserção da aresta mínima (Problema do Caixeiro Viajante).

A meta-heurística escolhida para o presente trabalho foi a busca tabu. Essa funciona da maneira que segue:

Uma solução heurística para o problema é encontrada e, sobre esta, aplicamos as operações de vizinhança, como em uma busca local. Entretanto, a busca tabu utiliza de um artifício para não causar um grande número de repetições nas soluções vizinhas encontradas (o que acontece com a busca local).

Esse artifício é a utilização de uma tabela, ou uma lista tabu, na qual são armazenados os movimentos já realizados recentemente, os quais serão "proibidos" de se repetirem. Assim, causamos uma diversificação e uma intensificação maior nos resultados.

2.2 A busca tabu aplicada ao PCV

O algoritmo implementado aplica a busca tabu ao Problema do Caixeiro Viajante. O PCV, como já citado anteriormente, é um problema NP-Difícil, portanto é interessante, para aproximarmos os resultados, o uso de meta-heurísticas.

A primeira parte a ser entendida é a solução inicial. Nesse projeto, a solução inicial foi obtida através da utilização da heurística do vizinho mais próximo para cada um dos vértices como inicial. Ou seja, utilizamos essa heurística n vezes, e encontramos, dentre os resultados, o de menor custo.

Após isso, temos a operação de vizinhança escolhida. Dada uma solução (representada por um vetor de inteiros - a sequência de vértices do ciclo), ela consiste em realizar permutações de modo que, inicialmente, o primeiro nó troque com outro se a solução gerada tiver custo menor do que a solução original (da qual queremos gerar o conjunto de vizinhos). Sendo assim, ao iniciar a operação temos um nodo de troca (V_k) setado para o primeiro nó do ciclo inicial. A partir do momento em que ocorre essa troca, setamos o novo nodo de troca (V_k) para o que trocamos com o inicial. Esse ciclo de trocas é repetido n vezes.

O algoritmo da busca tabu em si, gera a vizinhança da solução inicial e, para cada vizinho gerado, gera-se a vizinhança deste, selecionando das n vizinhanças, uma solução de menor custo.

A lista tabu vai incluir os movimentos de troca, com seus respectivos vértices. Por exemplo, ao trocarmos o vértice 3 com o 8, teremos, na lista tabu, o vetor [3,8], simbolizando esse movimento, que não poderá ser repetido até a lista atingir o seu tamanho máximo, quando o primeiro elemento da lista é retirado para a adição de outro no fim desta.

3 Resultados Computacionais

Os testes foram realizados com instâncias retiradas da *TSPLIB*. O algoritmo foi implementado em Java, em uma máquina Intel Core i7-4510U 2.0GHz, com 8GB de RAM. O sistema operacional no quais foram realizados os testes é um Fedora 22 de 64 bits.

A seguinte tabela apresenta a instância executada, o resultado exato da instância (R_{exato}), o resultado encontrado apenas com a heurística do vizinho mais próximo (R_{heur}), o resultado melhorado pela meta-heurística da busca tabu (R_{tabu}), a média do tempo de execução do algoritmo para 30 testes ($Media(T)$), e a variância do tempo ($Var(T)$).

Para os testes, foi assumido tamanho da tabela tabu igual a 5.

instância	R_{exato}	R_{heur}	R_{tabu}	$Media(T)(ms)$	$Var(T)$
eil51	460	816	718	164.96	70.21
a280	2579	4638	3343	62910.30	1540674.42
bier127	118282	230110	220241	2933.22	8177.80
eil76	583	1064	1021	506.19	300.93
eil101	629	1607	1385	1271.76	5464.36
gil262	2378	7411	7212	45364.02	5675037.82
kroA200	29368	101065	98633	22620.81	4819397.50
lin105	14379	28752	26115	1477.28	40056.62

Tabela 1: Resultados Computacionais

4 Conclusão

Sendo assim, com o presente trabalho, foram apresentados resultados esperados para uma meta-heurística, com um tempo computacional muito menor que o tempo para a execução de um algoritmo exato, entretanto, obtendo resultados bastante afastados da solução ótima.

Entendemos assim, que maior pesquisa na área de desenvolvimento de algoritmos é necessária, para a estimulação da criação de mais algoritmos eficientes, sejam aproximativos ou quicá exatos para problemas difíceis como o caixeiro viajante

Referências

- [1] Wikipedia. *Problema do Caixeiro-Viajante*, 2015. http://pt.wikipedia.org/wiki/Problema_do_caixeiro-viajante [Acessado em: 27/04/2015].
- [2] David L. [et al.] Applegate. The travelling salesman problem: a computational study. *Princeton: Princeton University Press*, 2006.
- [3] Richard M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103, 1972.
- [4] Leonora Bianchi; Marco Dorigo; Luca Maria Gambardella; Walter J. Gutjahr. A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing: an international journal*, 2009.