

UECE - CCT - Curso de Ciência da Computação

Trabalho da Disciplina: Teoria dos Grafos - 2016.2

"Algoritmo/Heurística para o Problema do Caixeiro viajante: *vizinho mais próximo com $n/3 \leq k \leq n/2$ partidas*"

Adson Roberto Pontes Damasceno

adson.uece@gmail.com

Matheus Lima Chagas

maths.c28@gmail.com

Anderson Bezerra Ribeiro

anderson0abr@gmail.com

22 de junho de 2017

Introdução

Neste artigo, apresentaremos uma implementação do algoritmo para o Problema do Caixeiro viajante utilizando a heurística do vizinho mais próximo com $n/3 \leq k \leq n/2$ partidas. Primeiramente descreveremos o problema e mostraremos o algoritmo utilizado. No final do artigo, mostramos e comentamos os resultados obtidos.

1 Descrição do Problema

Suponha um vendedor de produtos que atue em várias cidades, sendo que algumas delas são conectadas por estradas. O trabalho do vendedor exige que ele visite cada uma das cidades. É possível para ele planejar uma viagem de carro, partindo e voltando a uma mesma cidade, visitando cada uma delas exatamente uma vez? Se tal viagem for possível, é possível planejá-la de maneira a minimizar a distância total percorrida? Esse problema é conhecido como o Problema do Caixeiro viajante. Esse problema pode ser modelado como um grafo ponderado G , no qual os vértices correspondem a cidades e dois vértices estão unidos por uma aresta ponderada se e somente se as cidades correspondentes forem unidas por uma estrada, a qual não passa por nenhuma das outras cidades. O peso da aresta representa a distância da estrada entre as cidades. As perguntas propostas, para o novo contexto, são: o grafo G é um grafo hamiltoniano? Se for, é possível construir um ciclo hamiltoniano de peso (comprimento) mínimo?

1.1 Heurística Vizinho mais próximo com $n/3 \leq k \leq n/2$ partidas

Este algoritmo é utilizado para determinar uma solução para o problema do caixeiro viajante. Ele gera rapidamente um caminho curto, mas geralmente não o ideal.

Estes são os passos do algoritmo:

1. Escolha um vértice arbitrário como vértice atual.
2. Descubra a aresta de menor peso que seja conectada ao vértice atual e a um vértice não visitado V.
3. Faça o vértice atual ser V.
4. Marque V como visitado.
5. Se todos os vértices no domínio estiverem visitados, encerre o algoritmo.
6. Se não vá para o passo 2.

A sequência dos vértices visitados é a saída do algoritmo.

2 Resultados Obtidos

Tabela de Experimentos Computacionais

Tabela 1: Resultados obtidos – PCV*

N^o	Instância	n	Início	Máxima	Média	Solução	tempo(s)
1	Tsp10t3	10	9	145	145	145	15ms, 15ms, 11ms
2	Tsp12t2	12	1	42,06	41,93	41,68	7ms, 7ms, 6ms
3	Tsp16	16	10	72,85	64,88	54,40	8ms, 8ms, 9ms
4	Tsp20t3	20	2	1436	1407	1355	9ms, 10ms, 9ms
5	Tsp29t1	29	21	1970	1946,67	1935	17ms, 16ms, 14ms
6	Tsp58t1	58	5	27384	27384	27384	61ms, 69ms, 62ms
7	Tsp73t3	73	38	34614	34557,67	34445	172ms, 172ms, 139ms
8	Tsp225t2	225	207	4976,54	4927,31	4876,84	587ms, 514ms, 478ms
9	Tsp280t2	280	179	3275,30	3274,51	3273,77	748ms, 796ms, 918ms

2.1 Comentários sobre os resultados obtidos

O k utilizado no programa é um valor médio de $n/3$ e $n/2$. Observou-se que pelos custos, não houve uma disparidade entre os valores da máxima e da solução. Atribui-se que isso aconteceu devido ao fato do número de partidas ter sido suficiente para encontrar uma solução próxima da ótima, ao menos para essas instâncias. Ao analisar-se os números de vértices na tabela, percebe-se um grande salto na relação dos tempos de execução de certas instâncias para outras. Por exemplo: Da instância *Tsp20t3* para a instância *Tsp73t3*, o número de vértices da segunda é maior que o triplo da primeira. No entanto o tempo de execução com 73 vértices chega a ser quase 20 vezes o tempo da instância com 20 vértices. Assim, consegue-se observar o impacto que a complexidade $O(n^2)$ interfere no tempo de execução. Além disso, o algoritmo implementado, em casos de distâncias entre vizinhos equivalentes, prioriza o vizinho que foi comparado primeiramente, seguindo ordem numérica. Nesse caso, se for possível obter um ciclo de custo menor com os vizinhos desconsiderados não será possível obtê-lo.

3 Conclusão

Percebe-se que o algoritmo do vizinho mais próximo executa rapidamente, mas que acaba deixando passar rotas mais curtas. Devido a utilização de heurística gulosa, a rota encon-

trada é razoável, mas pode ser que não seja a ideal. Uma característica negativa do algoritmo está ligado ao fato das arestas, sendo estas localmente mínimas, a aresta final pode ser longa. Logo este algoritmo não encontra a solução ótima, a obtida está bem próxima do ótimo. Ainda assim podem ser encontradas instâncias em que a solução obtida pode ser muito ruim.

4 Referências Bibliográficas

1. CORMEN, Thomas H.

Algoritmos, Teoria e Prática;[tradução Arlette Simille Marques.] - Rio de Janeiro : Elsevier, 2012.il.

2. NICOLETTI, Maria do Carmo

Fundamentos da teoria dos grafos para computação / Maria do Carmo Nicoletti; Estevam Rafael Hruschka Jr.– São Carlos : EdUFSCar, 2006.

3. ZIVIANI, Nivio

Projeto de algoritmos com implementações Pascal e C / Nivio Ziviani.– 4. ad. – São Paulo : Pioneira, 1999. – (Pioneira Informática)