Analisando o Algoritmo de Dijkstra com Heap e sem Heap

Micael Andrade Dos Santos

[1]Departamento de Computação (DCOMP) – Universidade Federal de Sergipe (UFS)
Av. Marechal Rondon, s/n – Jardim Rosa Elze – CEP 49100-000
São Cristóvão – SE – Brazil

kaell_andrade@academico.ufs.br, micael.santos@dcomp.ufs.br

Resumo. *Este relatório tem como objetivo fazer uma análise simples das duas versões do algoritimo de Edsger Dijkstra para o cálculo de caminhos mínimos em grafos com arestas ponderadas com pesos positivos.*

1. Introdução

Há duas maneiras de implementar o Algoritmo de Dijkstra. Uma é usando busca sequencial, ou seja, para cada iteração do algoritmo será feita uma busca em todos os vértices do grafo para encontrar o menor vértice com tamanho mínimo que ainda não tenha sido visitado e, a outra é utilizando filas de prioridades (Heap).

2. Metodologia

Para a nálise foi utilizado grafos completos ponderados gerados de forma randômica: G1(100, 4950), G2(200, 19900), G3(500, 124750), G4(1000, 499500).

3. Estratégia de análise

Para cada um dos grafos listados anteiormente foi executado as duas versões do algorítimo e calculado seu determinado tempo, os tempos podem ser vistos na tabela a baixo.

Figura 1. Resultados

Grafo	*G1(100, 4950)*	*G2(200, 19900)*	*G3(500, 124750)*	*G4(1000, 499500*
Com Heap	0:00:00.283391	0:00:01.426038	0:00:21.602929	0:03:02.727487
Sem Heap	0:00:00.342469	0:00:02.788968	0:00:48.232816	0:05:25.666646

4. Resultados

Para cada caso foi utilizado o mesmo grafo e calculado o tempo. Como vimos anteriormente a implementação com Heap é bem mais eficiente do que sem a utilização da estrutura de prioridades. No primeiro o caso o algoritmo consome $O(n^2)$ unidades de tempo, já no segundo caso o tempo se resume a O(E lg V) caso todos os vértices possam ser alcançado da fonte. [Cormen et al. 2009]

Referências

[Cormen et al. 2009] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to algorithms*. MIT press.