AA de Grafos e Algoritmos
Algoritmo de Kruskal

Jamile Santos[1], **Liliane Neves**[1], **Victor Pedro**[1]

[1] Departamento de Ciência da Computação – Instituto Multidisciplinar
Universidade Federal Rural do Rio de Janeiro (UFRRJ)

{milysantos18,lilianeneves8}@gmail.com, victorp@r7.com

Resumo. *Este relatório descreve o estudo sobre como implementar um algoritmo capaz de separar um grafo valorado $G = (V, E)$ em k-grupos de vértices mais próximos. Para resolver este problema, o algoritmo deve encontrar uma árvore geradora mínima no grafo e após isso eliminar as $k - 1$ arestas. Para encontrar a árvore geradora mínima será utilizado o algoritmo de Kruskal e como complemento o algoritmo de busca em profundidade.*

1. Introdução

Dado um grafo não orientado conectado, uma árvore de dispersão deste grafo é um subgrafo, que é uma árvore que conecta todos os vértices.

Um único grafo pode ter diferentes árvores de extensão. Nós podemos assinalar um peso a cada aresta, que é um número que representa quão desfavorável ela é, e atribuir um peso a árvore de extensão calculado pela soma dos pesos das arestas que a compõem.

Uma árvore de extensão mínima (também conhecida como árvore de extensão de peso mínimo ou árvore geradora mínima) é então uma árvore de extensão com peso menor ou igual a cada uma das outras árvores de extensão possíveis. Generalizando, qualquer grafo não direcional (não necessariamente conectado) tem uma floresta de árvores mínimas, que é uma união de árvores de extensão mínimas de cada uma de suas componentes conexas.

Este trabalho objetiva dissertar sobre o uso do algoritmo de Kruskal no desafio de encontrar a Árvore Geradora Mínima de um grafo, assim como descrever o algoritmo e discutir sua complexidade.

2. Árvore Geradora Mínima

Este problema, como muitos semelhantes, tem seu foco em encontrar uma Árvore Geradora Mínima (MST - Minimum Spawning Tree) para um grafo conhecido.

Há uma solução genérica para este problema, que serve de base para o algoritmo de Kruskal, que é o discutido neste trabalho. Este algoritmo genérico mantém um conjunto de arestas A e a seguinte invariante:

A cada iteração, A é subconjunto de arestas de uma MST.

O algoritmo mantém esta invariante adicionando ao conjunto A, a cada iteração, uma aresta (u, v) de maneira que a união de A com (u, v) seja um subconjunto de uma MST [Feofiloff 2015].

```
GENERIC-MST(G,w)
1 A := {}
2 while A não forma uma árvore espalhada
3       do encontre uma aresta (u,v) que é segura para A
4          A := A união {(u,v)}
5 return A
```

Figura 1. Algoritmo MST Genérico

3. O Algoritmo de Kruskal

O algoritmo de Kruskal propõe que um grafo qualquer seja considerado uma floresta de árvores compostas por apenas um vértice e procura, a cada iteração, conectar duas árvores distintas da floresta através de uma aresta que possua peso mínimo.

Dessa forma, suponha que, numa determinada iteração, o algoritmo escolher a aresta (u, v) para ser inserida no conjunto A e que a aresta (u, v) conecte a árvore C_i à árvore C_2. Observe que, dentre todas as arestas que conectam duas componentes distintas nesta iteração, (u, v) é uma das que possui menor peso, pois foi escolhida. Logo, nesta iteração, para qualquer corte que separe u de v, (u, v) é uma aresta leve pelo fato de não existir aresta de menor peso separando duas componentes nesta iteração.

```
MST-KRUSKAL(G,w)
1 A := {}
2 for cada vértice v em V[G]
3     do MAKE-SET(v)
4 Ordene as arestas de E em ordem crescente de peso (w)
5 for cada aresta (u,v), tomadas em ordem crescente de peso (w)
6     do if FIND-SET(u) != FIND-SET(v)
7           then A := A união {(u,v)}
8                UNION(u,v)
9 return A
```

Figura 2. Algoritmo MST Kruskal

A Figura 3 exemplifica uma execução do algoritmo de Kruskal. É possível notar que as arestas são visitadas em ordem crescente de peso e, para que uma aresta seja inserida no conjunto A, seus vértices adjacentes devem pertencer a componentes diferentes.

4. Análise de Complexidade do Algoritmo

5. Conclusões

Como foi proposto, a implementação é capaz de separar o grafo em árvores conexas pela aresta de menor peso possível, utilizando uma Árvore Geradora Mínima através do algoritmo de Kruskal.

Os resultados obtidos pelo programa nos testes realizados apresentam sempre resultados corretos e tempos de execução pequenos, mesmo aumentando o número de vértices e arestas.

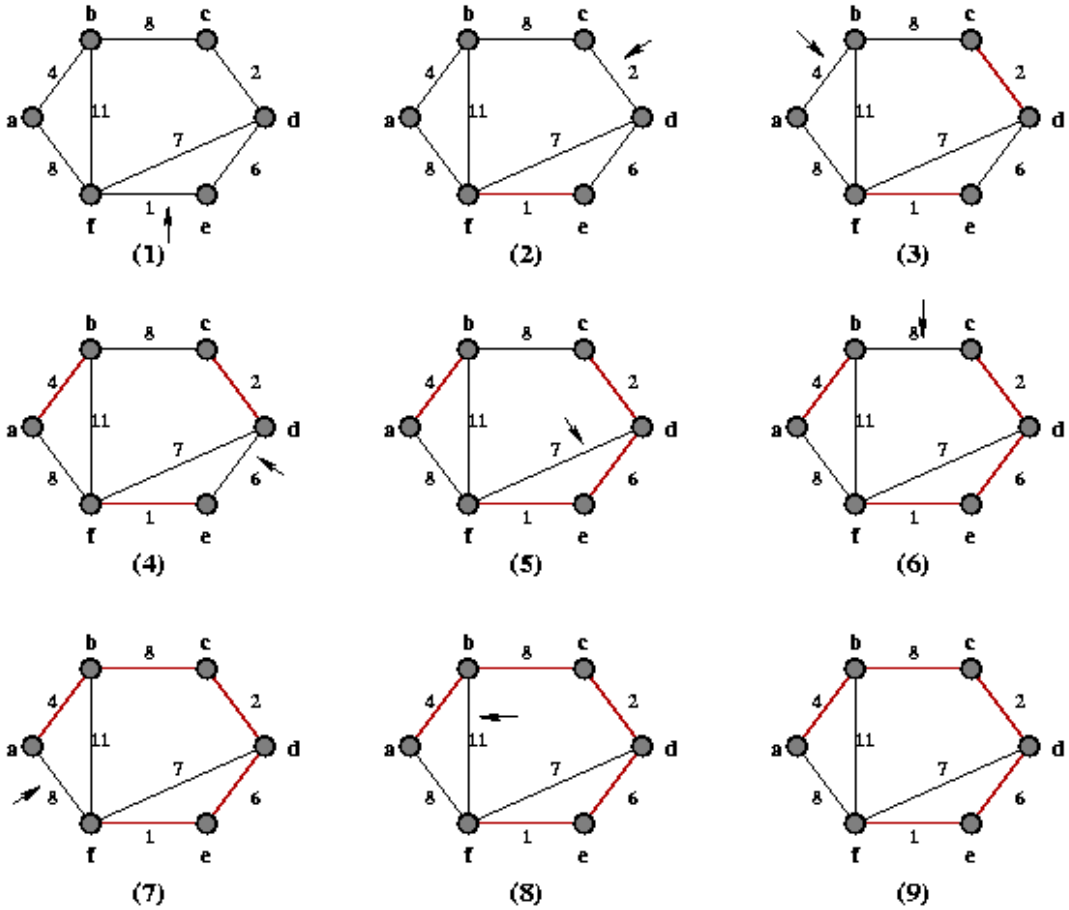

Figura 3. Algoritmo MST Genérico

Referências

Feofiloff, P. (2015). Algoritmo de kruskal. `http://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/kruskal.html`.

Thomas H. Cormen, Charles E. Leiserson, R. L. R. C. S. (2009). *Introduction to Algorithms*. MIT Press.