Algoritmos de Prim e Kruskal

Prim

O algoritmo de Prim constrói uma árvore geradora mínima por meio de uma sequência de subárvores em expansão. A subárvore inicial nessa sequência consiste em um único vértice selecionado arbitrariamente do conjunto V dos vértices do grafo.

Em cada iteração, o algoritmo expande a árvore atual de maneira gananciosa, anexando a ela o vértice mais próximo que não está naquela árvore. O algoritmo para depois que todos os vértices do grafo foram incluídos na árvore em construção.

Como o algoritmo expande uma árvore exatamente por um vértice em cada uma de suas iterações, ele é iterado n - 1, onde n é o número de vértices no grafo. A árvore gerada pelo algoritmo é obtida como o conjunto de arestas usadas para as expansões da árvore.

É necessário fornecer a cada vértice que não está na árvore atual, o nome do vértice da árvore mais próximo e o comprimento (o peso) da aresta correspondente. Vértices que não são adjacentes a nenhum dos vértices da árvore podem receber o rótulo ∞ indicando sua distância "infinita" aos vértices da árvore e um rótulo nulo para o nome do vértice da árvore mais próximo.

Com esses rótulos, encontrar o próximo vértice a ser adicionado à árvore atual T = (Vt, Et) torna-se uma tarefa simples de encontrar um vértice com o menor rótulo de distância no conjunto V - Vt. Empates podem ser quebrados arbitrariamente. Depois de identificarmos um vértice u⋆ a ser adicionado à árvore, precisamos realizar duas operações:

- Mover u⋆ do conjunto V - Vt para o conjunto de vértices da árvore Vt.
- Para cada vértice remanescente u em V - Vt que está conexo a u⋆ por uma aresta mais curta do que a atual de u, atualize seus rótulos por u⋆ e o peso da aresta entre u⋆ e u, respectivamente.

Árvore Geradora

Uma árvore geradora de um grafo não direcionado e conexo é uma árvore que contém todos os vértices do grafo. Se tal grafo tiver pesos atribuídos às suas arestas, uma árvore geradora mínima é a sua árvore geradora de menor peso, onde o peso de uma árvore é definido como a soma dos pesos em todas as suas arestas.

O problema da árvore geradora mínima é o problema de encontrar uma árvore geradora mínima para um dado grafo conexo ponderado.

Kruskal

O algoritmo de Kruskal considera uma árvore geradora mínima de um grafo ponderado conexo G = (V, E) como um subgrafo acíclico com |V| - 1 arestas, o qual a soma dos pesos das arestas é a menor possível.

Consequentemente, uma árvore geradora mínima é construída como uma sequência em expansão de subgrafos acíclicos, mas não necessariamente conexos nas etapas intermediárias do algoritmo.

O algoritmo começa ordenando as arestas do grafo em ordem crescente de seus pesos. Então, começando com o subgrafo vazio, ele percorre essa lista ordenada, adicionando a próxima aresta na lista ao subgrafo atual se essa inclusão não criar um ciclo e pula a aresta caso contrário.

Subsets Disjuntos e Algoritmos Union-Find

Há 2 principais formas de implementar essa estrutura de dados. Quick find, otimiza a eficiência temporal da operação find; a segunda, chamada de quick union, otimiza a operação de union.

Após ser inicializada como uma coleção de n subconjuntos com 1 elemento, distintos de S, a coleção é submetida a uma sequência de operações de união e busca intercaladas. Assim, estamos lidando com as seguintes operações:

- makeset(x): cria um conjunto de um único elemento {x}. Assume-se que esta operação pode ser aplicada a cada um dos elementos do conjunto S apenas uma vez.

- find(x) retorna um subconjunto contendo x.
- union(x, y) constrói a união dos subconjuntos disjuntos Sx e Sy contendo x e y, respectivamente, e a adiciona à coleção para substituir Sx e Sy, que são excluídos dela.