PRIM`S ALGORITHM

ÁRVORE GERADORA

A árvore geradora de um grafo conectado não direcionado é seu subgrafo acíclico conectado (ou seja, uma árvore) que contém todos os vértices do grafo. Se tal grafo tem pesos atribuídos às suas arestas, uma árvore geradora mínima é sua árvore geradora de menor peso, onde o peso de uma árvore é definido como a soma dos pesos em todas as suas arestas.

FUNCIONAMENTO

O algoritmo de Prim constrói uma árvore geradora mínima através de uma sequência de expandir subárvores. A subárvore inicial em tal 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, simplesmente anexando a é o vértice mais próximo que não está naquela árvore.O algoritmo para após todos os vértices do grafo sejam incluídos na árvore que está sendo construída. Como o algoritmo expande uma árvore por exatamente um vértice em cada uma de suas iterações, o número total de tais iterações é n − 1, onde n é o número de vértices no gráfico.

EFICIÊNCIA

Se um grafo é representado por sua matriz de peso e a fila de prioridade são implementadas como uma matriz não ordenada, o tempo de execução do algoritmo será em (|V | 2).Se um grafo é representado por suas listas de adjacências e a fila de prioridade é implementada como um min-heap, o tempo de execução do algoritmo é O(|E| log |V |).

UTILIZAÇÕES

Tem aplicações diretas no projeto de todos os tipos de redes— incluindo comunicação, computador, transporte e eletricidade - fornecendo a maneira mais barata de obter conectividade. Ele identifica grupos de pontos em conjuntos de dados. Tem sido usado para fins de classificação em arqueologia, biologia, sociologia e outras ciências. Também é útil para construir soluções aproximadas para mais problemas difíceis como o problema do caixeiro viajante

KRUSKAL`S ALGORITHM

KRUSKAL VS PRIM

Aplicando manualmente os algoritmos de Prim e Kruskal ao mesmo pequeno grafo
pode criar a impressão de que o último é mais simples que o primeiro. Essa impressão está errada porque, a cada uma de suas iterações, o algoritmo de Kruskal precisa verificar
se a adição da próxima aresta às arestas já selecionadas criaria um ciclo.

FUNCIONAMENTO

O algoritmo de Kruskal analisa um
árvore geradora mínima de um grafo conectado ponderado G = V,E como um acíclico
subgrafo com |V | − 1 arestas para as quais a soma dos pesos das arestas é a menor.
(Não é difícil provar que tal subgrafo deve ser uma árvore.) Conseqüentemente, o algoritmo constrói uma árvore geradora mínima como uma sequência em expansão de
subgrafos que são sempre acíclicos, mas não necessariamente conectados nos estágios intermediários do algoritmo.
O algoritmo começa classificando as arestas do grafo em ordem não decrescente de
seus pesos. Então, começando com o subgrafo vazio, ele varre esta lista ordenada, adicionando a próxima aresta na lista ao subgrafo atual se tal inclusão não
não crie um ciclo e simplesmente pule a borda de outra forma.

INTERPRETAÇÃO

Podemos considerar as operações do algoritmo
como uma progressão através de uma série de florestas contendo todos os vértices de um dado
gráfico e algumas de suas arestas. A floresta inicial consiste em |V | árvores triviais, cada
compreendendo um único vértice do grafo. A floresta final consiste em uma única árvore,
que é uma árvore geradora mínima do grafo. A cada iteração, o algoritmo
pega a próxima aresta (u, v) da lista ordenada das arestas do grafo, encontra as árvores
contendo os vértices u e v, e, se essas árvores não forem iguais, une-as
em uma árvore maior adicionando a aresta (u, v).

DISJOINT SUBSETS/UNION FIND

O algoritmo de Kruskal é uma das várias aplicações que requerem uma dinâmica
partição de algum conjunto de n elementos S em uma coleção de subconjuntos disjuntos S1, S2,...,Sk. A maioria das implementações desse tipo de dado abstrato usa um elemento de cada um
os subconjuntos disjuntos em uma coleção como o representante desse subconjunto. Aqui estão duas alternativas principais para implementar essa estrutura de dados. O
o primeiro, chamado de busca rápida, otimiza a eficiência de tempo da operação de busca;
a segunda, chamada de união rápida, otimiza a operação de união.