Problem Solving Techniques and Graph Theory

Programming Club

(2016-17)

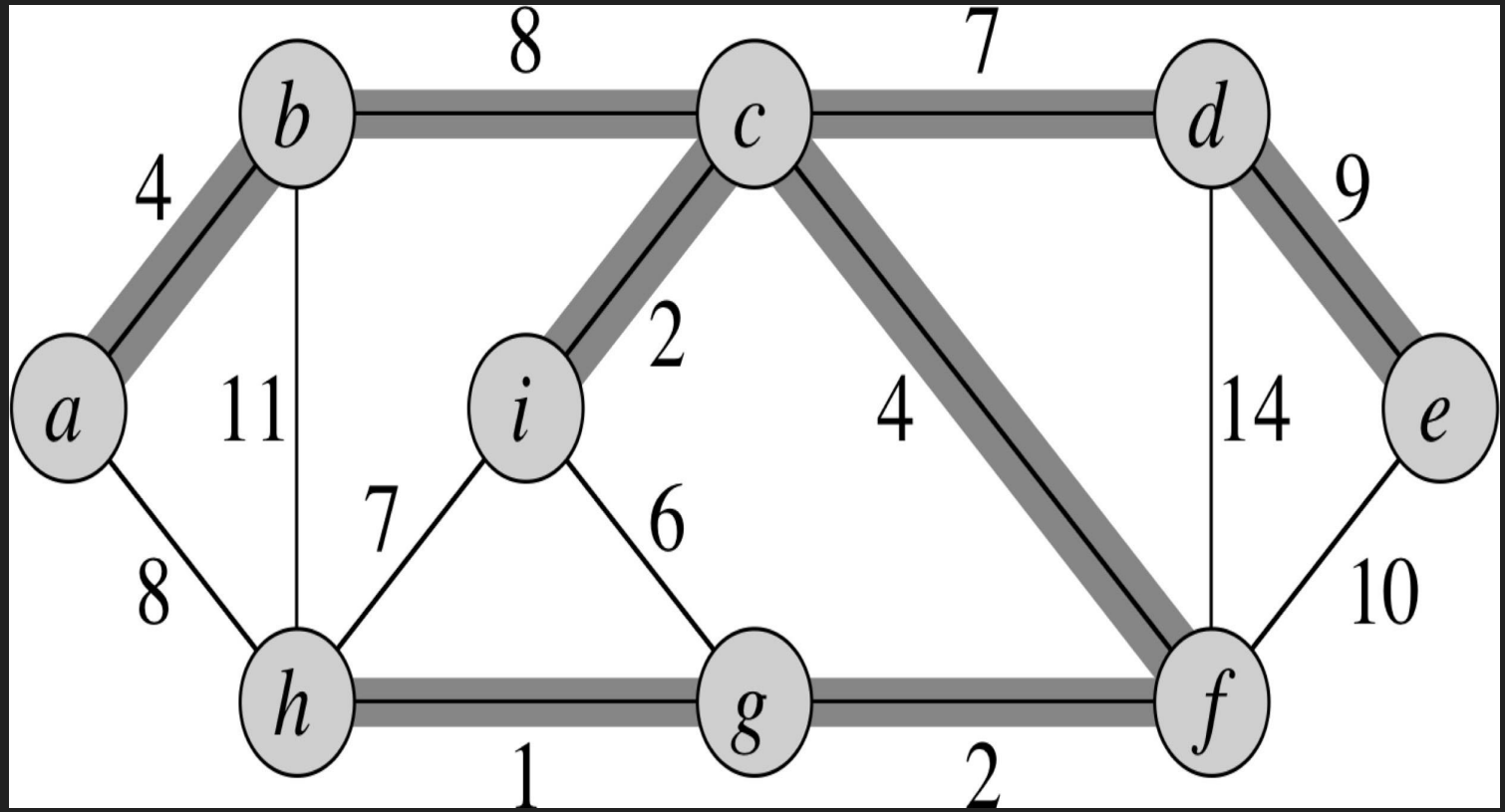
Science and Technology Council

IIT Kanpur

Minimum Spanning Tree

- Given a graph, find a tree such with the least sum of edges
- Many applications like network design
- Algorithms -
  - Kruskal’s Algorithm
  - Prim’s Algorithm

How will you solve it?



Prim’s Algorithm

- Start with an arbitrary vertex
- Which vertex will you pick next - which one is sure to be included?

Prim’s Algorithm

- Store the distances from this vertex to all vertices
- Take the vertex with minimum distance and remove it from the list
- Now for each of its neighbouring vertices (of the vertex picked just now), update the distance as the min(current_distance, distance from this vertex)
- Again pick the one with minimum distance till all vertices are picked

Prim’s Algorithm (Time complexity)

- Store the distances from this vertex to all vertices (O(V) time)
- Take the vertex with minimum distance and remove it from the list (O(V) time)
- Now for each of its neighbouring vertices (of the vertex picked just now), update the distance as the min(current_distance, distance from this vertex). This takes time equal to the number of edges as updating will take constant time. On summing, it will take O(E) time total
- Again pick the one with minimum distance till all vertices are picked
- These steps will be repeated V times so it will take O(V^2) time

Problem Solving

Gold and the brothers

Your father has a gold mine in the form of an NxN grid. Each square in the grid has some (not equal) gold deposits. The father is on his deathbed, and has decided to divide the wealth amongst his 4 sons, you being the youngest one. To divide his wealth, he will draw a horizontal line and a vertical line in the grid, which will divide his property in 4 rectangles. Your brothers will choose 3 out of the 4 smaller fields, and you will get the last one. (You won't be able to argue with them on this since you don't stand a chance against your elder brothers)

You are the smartest kid, and your father has asked you to divide the field for him. How will you divide the field so that you get the maximum possible gold, assuming your brothers will pick the best 3 rectangles.

Maximum Sub Subsection

Let’s not waste our time with the story and get on to the question directly from now onwards.

Given a sequence of integers, a subsection is a block of contiguous integers in the sequence. Your task is to find the subsection with the maximum sum.

Buying empty plots

I guess it gets a bit boring without the story. So here comes another story.

Now you have a lot of gold (remember?) and have decided to buy empty plot to start a business. Along one side of the road next to your college is a sequence of vacant plots of land. Each plot has a different area (>0) and they form a sequence A[1], A[2], ... A[N]

You want to buy K acres of land, so you want to find a segment of contiguous plots (subsection) of minimum length, whose sum of area is exactly K acres.

It is given that such a sequence exists for sure.

Buying TWO empty plots

Because of your brilliance, you excelled at your business, and now you have a lot of money. You discover a similar sequence of plots as described in the last problem, this time near the place where your childhood crush lives now. To show off in front of them, you decide to buy two plots this time, having an area of K acres each, such that the number of plots in the two subsections is minimised.

Note: The two subsections must be disjoint.

Happily married

The trick worked, and now you are happily married. You have set up a beautiful garden, arranged as a rectangular grid of squares. Each square contains a number of roses. Since you spent most of your money buying the two plots, you can only afford two gardeners to take care of those roses now. Each gardener will take care of exactly K roses, where K is a fixed number as before. Each gardener want to work on a rectangular patch. To avoid the gardeners interfering with each other, the two patches must be disjoint (though they may share a boundary). The gardener charges a fee, which is equal to the cost of the patch he has to maintain.

Your aim is to find two non overlapping patches for the gardeners, each containing exactly K roses, so that you have to pay the least salary to those gardeners. If the two gardeners share boundary, that perimeter is counted in the perimeter of both the rectangles. Here no. of rows and columns <= 250, K<=2500, no. of roses <=5000

Bathroom Designing: 1

Now that your garden is taken care of, you have decided to tile your bathroom walls. You have an (nx2) grid, and an infinite supply of rectangular tiles of dimension 2x1. The tiles can be laid down horizontally or vertically both.

How many ways are there to tile?

Bathroom Designing: 2

The scenario is exactly same as that of the last problem, except for the fact that we have one more kind of tile now, an L shaped tile, like this:.

Now, how many different ways are there to tile the bathroom?

Counting paths

I finally ran out of ideas for proceeding with the story. Let’s get back to business.

You have a rectangular grid of points with N rows and N columns, You start at the top left corner, and at each step, you can either go right, or down. You need to reach the bottom right corner.

How many paths are there from the top left to the bottom right?

Counting paths

The last question was easy, right?

Now let’s modify it so you can really appreciate dynamic programming.

What if some cells are blocked, i.e. you can’t step on those cells?

What if you can only go at most K cells continuously in one direction?

Longest Common Subword

You are given two words, for example:

V: ABCBABDB

W: ABDBABBC

Your task is to find out the longest contiguous sequence of letters (subword), that is common to both the strings.

Longest common subsequence

You are given two words as before, but here, instead of finding the longest common subword, you need to find out the longest common subsequence. A subsequence is different from a subword on the terms that you are allowed to drop a few letters, and then obtain a sequence.

For ex. KNL is a subsequence of KUNAL.

Homework:

We just discussed the algorithms to compute the length of the longest common subword and subsequence in two strings.

Unfortunately, these algorithms only give out the length, but not the longest subsequence/subword

Modify the algorithms to find out those words.

An interesting problem

You are given a graph as before, and here you have to find out the shortest distance between all pair of vertices.

How will you do it?

You can surely solve this on your own!

Floyd Warshall’s Algorithm (continued)

- Naive solution takes exponential time
- We shall try to use a property:
- If the shortest path between i and j includes k, then the shortest path is composed of the shortest path between i and k, and the shortest path between k and j

Floyd Warshall’s Algorithm (continued)

- Find all pairs of shortest distances using vertices lying between only 1 and n as intermediate vertices
- Use recursion
- f(i,j,n) - shortest distances between i and j using vertices lying between only 1 and n as intermediate vertices
- Can you break this into a subproblem?

Floyd Warshall’s Algorithm (recursive)

- We know f(i,j,0) - it is the original matrix
- To express f(i,j,k) in terms of f(i,j,k-1)
- Either we use k as an intermediate vertex or we do not use it
  - If we use k as intermediate vertex
  - f(i,j,k) = f(i,k,k-1) + f(k,j,k-1) using property and the fact that k will not be used as an intermediate matrix
  - If we do not use k as an intermediate vertex
  - f(i,j,k) = f(i,j,k-1)
- We take min of these two to get the shortest path
- This is the recursive approach

Floyd Warshall’s Algorithm (recursive)

- We shall start bottom up
- We already have the matrix for f(i,j,0)
- Using it, we find the matrix for f(i,j,1) and so on
- Our answer is f(i,j,V)
- f(i,j,k) = max (f(i,j,k-1), f(i,k,k-1) + f(t,k,k-1))
- This step takes O(V^2) time
- Since we construct V such matrices, time complexity is O(V^3)

Floyd Warshall's Algorithm (pseudocode)

```
FLOYD-WARSHALL(W)
1. n ← rows[W]
2. D^(0) ← W
3. for k ← 1 to n
4.     do for i ← 1 to n
5.         do for j ← 1 to n
6.             d_ij^(k) ← min(d_ij^(k-1), d_ik^(k-1) + d_kj^(k-1))
7. return D^(n)
```

Prim’s algorithm using Priority queue

Priority queue

- It is a data structure which returns the minimum of the elements stored in it in log(n) time
- Updating a value in it also takes log(n) time
- You can read about its implementation

Prim’s Algorithm (Priority Queue)

- Store the distances from this vertex to all vertices in a priority queue
- Take the vertex with minimum distance and remove it from the priority queue
- Now for each of its neighbouring vertices (of the vertex picked just now), update the distance as the min(current_distance, distance from this vertex)
- Again pick the one with minimum distance till all vertices are picked

Prim’s Algorithm (priority queue - time complexity)

- Store the distances from this vertex to all vertices in a heap (takes O(V) time)
- Take the vertex with minimum distance and remove it from the heap (takes O(log V) time)
- Now for each of its neighbouring vertices (of the vertex picked just now), update the distance as the min(current_distance, distance from this vertex). This takes O(log V) time in the PQ. Since this will be repeated once for each edge in the graph, it will take O(E log V) time
- Again pick the one with minimum distance till all vertices are picked
- Total time will be O(V) + O(V log V) + O(E log V) = O(E log V)

Prim's Algorithm (Pseudo code)

```
MST-PRIM(G, w, r)
 1  for each u ∈ G.V
 2      u.key = ∞
 3      u.π = NIL
 4  r.key = 0
 5  Q = G.V
 6  while Q ≠ ∅
 7      u = EXTRACT-MIN(Q)
 8      for each v ∈ G.Adj[u]
 9          if v ∈ Q and w(u, v) < v.key
10              v.π = u
11              v.key = w(u, v)
```

Thank you

For any queries, contact us on Facebook or via email.