Algorithms and Data Structures

Exercise Session 11

https://n.ethz.ch/~ahmala/and

Maximum Apples in a Grid [DP]

You are given an N x M grid where each cell contains a certain number of apples. You start at the top-left corner of the grid and can only move right or down.

Determine the maximum number of apples you can collect by the time you reach the bottom-right corner of the grid.

Solve in O(N x M) time.

Key Observations

- You can reach any cell (i, j) from:
  - The left cell (i, j - 1) if j > 0
  - The top cell (i - 1, j) if i > 0

Key Observations

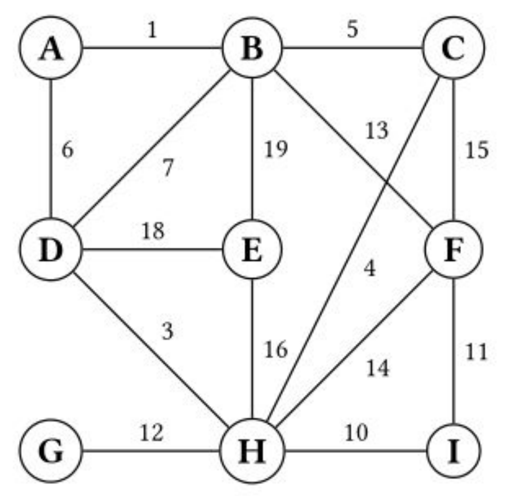
- You can reach any cell (i, j) from:
  - The left cell (i, j - 1) if j > 0
  - The top cell (i - 1, j) if i > 0
- To maximize apples in cell (i, j), you take the maximum apples collected from either path and add the apples in the current cell.

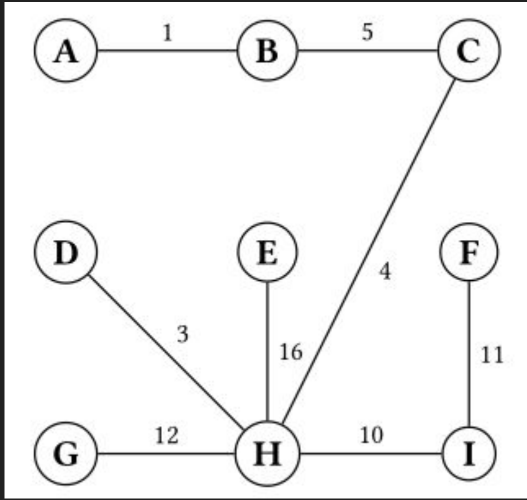
Recurrence Formula

- $S[i][j] = A[i][j] + \max(S[i-1][j], S[i][j-1])$
  - If i = 0, ignore S[i-1][j].
  - If j = 0, ignore S[i][j-1].

Minimum Spanning Tree

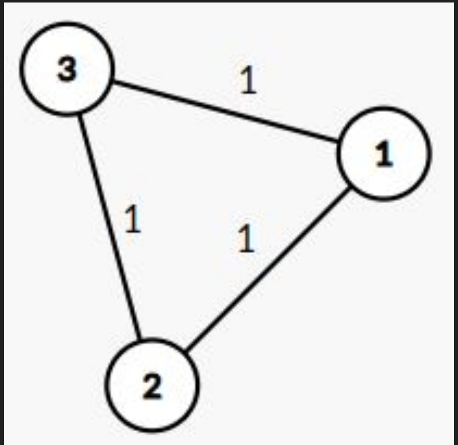
subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight





Uniqueness of MST

If each edge has a distinct weight then there will be only one, unique minimum spanning tree



Boruvka

- Initialize all vertices as individual components (or sets).
- Initialize MST as empty.
- While there are more than one component, do the following for each component:
  - Find the closest weight edge that connects this component to any other component.
  - Add this closest edge to MST if not already added.
- Return MST.

Prim's Algorithm

- Initialize
  - Start with an arbitrary vertex, mark it as part of the MST.
  - Initialize the MST as empty.
  - Maintain a priority queue (or min-heap) to store edges with their weights, starting with edges connected to the initial vertex.
- While there are vertices not in the MST:
  - Extract the edge with the smallest weight from the priority queue that connects a vertex in the MST to one outside.
  - Add the vertex from this edge to the MST.
  - Add all new edges connecting the newly added vertex to vertices outside the MST to the priority queue.
- Return the MST when all vertices are included.

Kruskal’s Algorithm

- Initialize:
  - Sort all the edges of the graph in increasing order of their weights.
  - Initialize a disjoint-set (or union-find) data structure to keep track of components (sets).
- While there are edges left:
  - Take the smallest edge from the sorted list.
  - Check if the two vertices of the edge are in the same component (set) using the disjoint-set.
  - If they are in different components, add the edge to the MST and union the two components.
  - If they are in the same component, discard the edge (it would form a cycle).
- Return the MST when the number of edges in the MST equals V-1 (where V is the number of vertices).

In Prim’s algorithm the edges are added in the following order: {G, H}, {D, H}, {C, H}, {B, C},{A, B}, {H, I}, {F, I} and {E, H}.

In Kruskal’s algorithm the edges are added in the following order: {A, B}, {D, H}, {C, H}, {B, C},{H, I}, {F, I}, {G, H} and {E, H}.

In boruvka in the first step we add the edges {A, B} (for A and B), {C, H} (for C), {D, H} (for D and H),

{E, H} (for E), {F, I} (for F), {G, H} (for G) and {H, I} (for I). This gives the components

{A, B} and {C, D, E, F, G, H, I}. In the second step, we add the edge {B, C}. Thus the minimum

spanning tree consists of the following edges:

Peer Grading

Exercise 10.4