

Basic Graph Algorithms

Thursday, September 28, 2017 5:03 PM

Problem A: Arctic Network

Problem B: Island Hopping

- Minimum Spanning Tree

A **minimum spanning tree (MST)** or **minimum weight spanning tree** is a 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.

[Minimum spanning tree - Wikipedia](https://en.wikipedia.org/wiki/Minimum_spanning_tree)
https://en.wikipedia.org/wiki/Minimum_spanning_tree

- Prim's Algorithm

Prim's Algorithm

<p>1 Given a network.....</p>	<p>2 Choose a vertex</p>	<p>3 Choose the shortest edge from this vertex.</p>
<p>4 Choose the nearest vertex not yet in the solution.</p>	<p>5 Choose the next nearest vertex not yet in the solution, when there is a choice, choose either.</p>	<p>6 Repeat until you have a minimal spanning tree.</p>

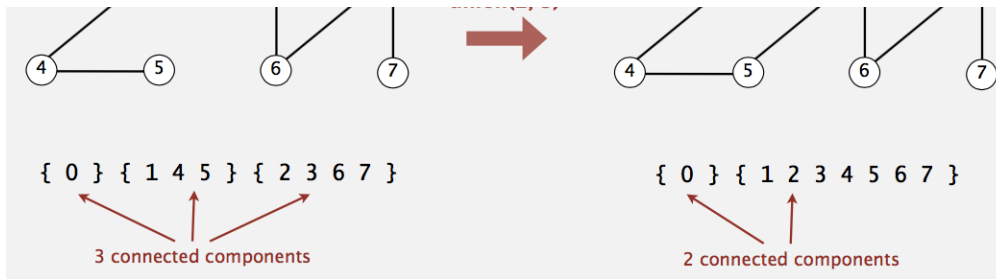
- Kruskal's Algorithm

Kruskal's Algorithm

<p>1 Given a network.....</p>	<p>2 Choose the shortest edge (if there is more than one, choose any of the shortest).....</p>	<p>3 Choose the next shortest edge and add it.....</p>
<p>4 Choose the next shortest edge which wouldn't create a cycle and add it.</p>	<p>5 Choose the next shortest edge which wouldn't create a cycle and add it.</p>	<p>6 Repeat until you have a minimal spanning tree.</p>

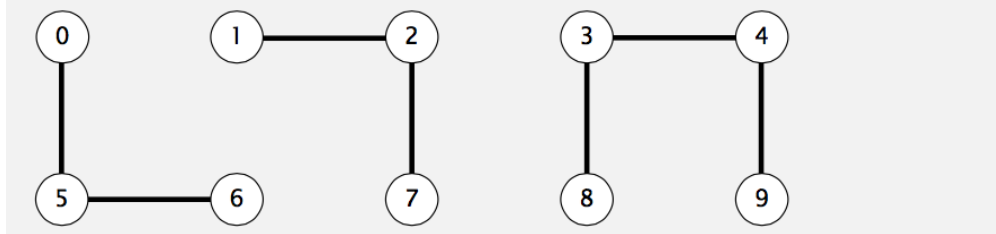
- Disjoint-set data structure / Union-find data structure





	0	1	2	3	4	5	6	7	8	9	
id[]	0	1	1	8	8	0	0	1	8	8	

0, 5 and 6 are connected
 1, 2, and 7 are connected
 3, 4, 8, and 9 are connected



Union

3-4	0	1	2	4	4	5	6	7	8	9
4-9	0	1	2	9	9	5	6	7	8	9
8-0	0	1	2	9	9	5	6	7	0	9
2-3	0	1	9	9	9	5	6	7	0	9
5-6	0	1	9	9	9	6	6	7	0	9
5-9	0	1	9	9	9	9	9	7	0	9
7-3	0	1	9	9	9	9	9	9	0	9
4-8	0	1	0	0	0	0	0	0	0	0
6-1	1	1	1	1	1	1	1	1	1	1

problem: many values can change

• Reference

- https://en.wikipedia.org/wiki/Minimum_spanning_tree
- <https://www.hackerearth.com/practice/algorithms/graphs/minimum-spanning-tree/tutorial/>

- <http://www.geeksforgeeks.org/greedy-algorithms-set-2-kruskals-minimum-spanning-tree-mst/>
- https://en.wikipedia.org/wiki/Prim%27s_algorithm
- https://en.wikipedia.org/wiki/Kruskal%27s_algorithm
- https://en.wikipedia.org/wiki/Disjoint-set_data_structure
- <http://vancexu.github.io/2015/07/13/intro-to-union-find-data-structure.html>

Problem C: Nikola

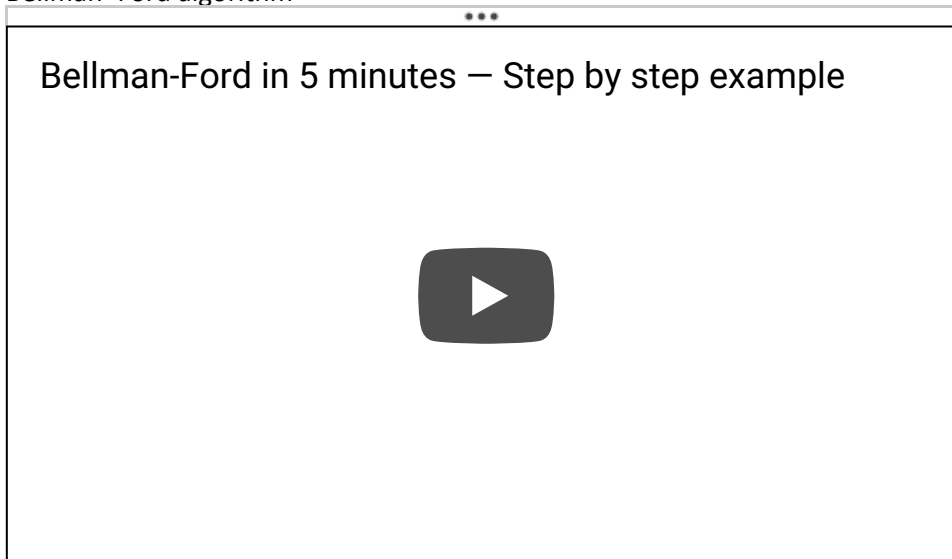
- Solved by Ziyi Zhang 🙌🙌🙌
- | | | |from| | |...| | |i| | |...| | |to| | | |
- Where from = $i - (\text{stepLength} - 1)$, to = $i + \text{stepLength}$
- $\text{mincost}[i][j] = \min(\text{mincost}[i + \text{stepLength}][\text{stepLength}], \text{mincost}[i - (\text{stepLength} - 1)][\text{stepLength}])$
- Time complexity: $\Theta(n^2)$

Problem D: Reachable Roads

- See "Disjoint-set data structure / Union-find data structure"

Problem E: Single source shortest path, negative weights

- Bellman-Ford algorithm



- Reference
 - https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm
 - https://www-m9.ma.tum.de/graph-algorithms/spp-bellman-ford/index_en.html (Highly recommended)
- Our Github repository:
 - <https://github.com/atmorgan/ICPC2014/blob/master/BellmanFord.cc>