Competitive Programming and Mathematics Society

Programming Workshop #3

Shortest Path Problems

Patrick Moore and Ryan Ong

Today's Workshop

CPMSOC

1 All-Pairs Shortest Path Problem

2 Floyd-Warshall Algorithm

3 Bellman-Ford Algorithm

4 Problem: Arbitrage

5 Problem: Heavy Flies

6 Wrap up

All-Pairs Shortest Path Problem

You are given a graph G with N nodes and M weighted directed edges. Edge weights may be negative. Find the shortest distance between all pairs of nodes in G.

Floyd-Washall Algorithm

CPMSOC

initialise an adjacency matrix dist[][] as follows for all i and j:

if there is an edge from i to j:

- dist[i][j] is the weight of the edge

if $i == j$:

- dist[i][j] is 0

otherwise dist[i][j] is infinity

Floyd-Washall Algorithm

CPMSOC

initialise an adjacency matrix dist[][] as follows for all i and j:
if there is an edge from i to j:

- dist[i][j] is the weight of the edge

if $i == j$:

- dist[i][j] is 0

otherwise dist[i][j] is infinity

for k from 1 to N:

- for i from 1 to N:
  - for j from 1 to N:
    - dist[i][j] = min(dist[i][j], dist[i][k] + dist[k][j])

Analysis of Floyd-Warshall

CPMSOC

Floyd-Warshall runs in $O(N^3)$.

Analysis of Floyd-Warshall

Floyd-Warshall runs in O(N^3).

If the edge weights are non-negative, then you can use Dijkstra's Algorithm for single source shortest paths for O($N * M * log(N)$).

Behaviour with negative weights

CPMSOC

The Floyd-Warshall Algorithm performs perfectly fine with negative weights!

Definition

A negative weight cycle occurs when in which you can begin at a node X, take some path around the graph and back to X such that the sum of the weights on the graph is negative.

Negative weight cycles break shortest-path algorithms, but we can detect such cycles by checking the dist[i][i] for all i from 1 to N and seeing if they are negative.

Single Source Shortest Path Problem

CPMSOC

You are given a graph G with N nodes and M edges. Edge weights may be negative. You are also given a source, S. You must find the minimum distance from S to all nodes in the graph.

Note that since edge weights may be negative, Dijkstra's algorithm will not work.

Bellman-Ford Algorithm

CPMSOC

Create arrays distance[V], initialised to infinity (except distance[S] = 0) and parent[V], initialised to null.

Bellman-Ford Algorithm

CPMSOC

Create arrays distance[V], initialised to infinity (except distance[S] = 0) and parent[V], initialised to null.

for i from 1 to $V - 1$:

- for each edge (u -> v; w):
  - if distance[u] + w < distance[v]:
    - distance[v] = distance[u] + w
    - parent[v] = u

Bellman-Ford Algorithm

CPMSOC

Create arrays distance[V], initialised to infinity (except distance[S] = 0) and parent[V], initialised to null.

for i from 1 to $V - 1$:

- for each edge (u -> v; w):
  - if distance[u] + w < distance[v]:
    - distance[v] = distance[u] + w
    - parent[v] = u

To detect and report the cycle, repeat the inner loop one more time. If there is any change, then there must be a negative weight cycle. Follow the trail of edges that improve the results and

Analysis of Bellman-Ford

CPMSOC

Bellman-Ford runs in O($N * M$).

Analysis of Bellman-Ford

Bellman-Ford runs in O($N * M$).

Several constant-factor optimisations exist for Bellman-Ford, generally by tweaking the order in which edges are visited to make updates propagate faster. It is possible to reduce the repetitions of the outer loop to $N/2$ in the worst case, or $N/3$ on average. While beneficial in some cases, these generally aren't necessary in competitions.

Problem: Arbitrage

CPMSOC

Arbitrages use the exchange rates between currencies of different exchanges to turn 1 unit of a currency into more than 1 unit of a currency.

Given a set of directed exchange rates between different currencies, determine if an arbitrage is possible.

Problem: Heavy Flies

CPMSOC

You are given an undirected weighted graph G with N nodes and M edges. You are also given a source S and a destination T. You need to output the shortest path from S to T.

Problem: Heavy Flies

CPMSOC

You are given an undirected weighted graph G with N nodes and M edges. You are also given a source S and a destination T. You need to output the shortest path from S to T.You also need to output the *second* shortest path from S to T. You are guaranteed that there is only 1 shortest path from S to T.

Attendance

CPMSOC

https://forms.gle/jaohN8kE4yTimY9y5

Wrap up

CPMSOC

- Problems:
  - Implement Floyd-Warshall or Bellman-Ford and compare its performance to Dijkstra on graphs with positive edge weights
  - Arbitrage (SPOJ): https://www.spoj.com/problems/ARBITRAG/
  - Heavy Flies
  - Tourist Guide (UVA 10099) https://onlinejudge.org/index.php?option=com$_onlinejudgeItemid = 8page = show_problemproblem = 1040$
  - Greg and Graph: https://codeforces.com/contest/295/problem/B
- CP workshops will be held in weeks 3, 5 and 7, probably same time and place.
- A reminder about the competitive maths workshops that run in weeks 2, 4, 6 8.