Graph Algorithms

15-121

Introduction to Data Structures

Ananda Gunawardena

7/31/2011
In this lecture..

- Main idea is finding the **Shortest Path between two points in a Graph**

- We will look at
  - Graphs with non negative cost edges
    - Dijkstra’s Algorithm
Shortest Paths
Airline routes
How to find the shortest Path?

The naïve solution is $O(n!)$
Greedy Algorithms
Greedy Algorithms

- In a *greedy algorithm*, during each phase, a decision is made that appears to be optimal, without regard for future consequences.
- This “take what you can get now” strategy is the source of the name for this class of algorithms.
- When a problem can be solved with a greedy algorithm, we are usually quite happy.
- Greedy algorithms often match our intuition and make for relatively painless coding.
Greedy Algorithms

- 4 ingredients needed
  - Optimization problem
    - Maximization or minimization
  - Can only proceed in stages
    - No direct solution available
  - Greedy Choice Property
    - A locally optimal solution (greedy) will lead to a globally optimal solution
  - Optimal Substructure
    - An optimal solution to the problem contains, within it the optimal solution to the sub problem
Examples

- Find the minimum number of coins necessary to change 63 cents
  
  - Assume we have 25-cent, 10-cent, 5-cent, 1-cent coins

- Dijkstra’s algorithm for shortest paths
  
  - Next...
Shortest Path Algorithm
for
Non-negative weights
(Dijkistra’s Algorithm)
Weighted shortest path

- Now suppose we want to minimize the total mileage.

- Breadth-first search does not work!
  - Minimum number of hops does not mean minimum distance.
  - Consider, for example, BWI-to-DFW:
Three 2-hop routes to DFW

1. SFO to ORD to DFW
   - SFO: 337
   - ORD: 1846
   - DFW: 802

2. LAX to JFK to DFW
   - LAX: 1235
   - JFK: 1391
   - DFW: 1391

3. BOS to BWI to DFW
   - BOS: 2342
   - BWI: 946
   - DFW: 1121

Distances in miles:
- SFO to ORD: 1846
- ORD to JFK: 849
- JFK to DFW: 621
- SFO to JFK: 2704
- ORD to DFW: 2704
- JFK to DFW: 740
- ORD to BWI: 867
- BWI to DFW: 187
- BOS to DFW: 2342
- BWI to MIA: 2342
- MIA to DFW: 1090
- LAX to JFK: 849
- JFK to MIA: 621
- MIA to DFW: 1121
- BOS to JFK: 144
- JFK to MIA: 144
- MIA to DFW: 946
- BOS to LAX: 1258
- LAX to DFW: 1258
- DFW to MIA: 2342
- MIA to SFO: 802
Dijkstra’s Algorithm
Intuition behind Dijkstra’s alg.

- For our airline-mileage problem, we can start by guessing that every city is $\infty$ miles away.
  
  ➢ Mark each city with this guess.

- Find all cities one hop away from BWI, and check whether the mileage is less than what is currently marked for that city.
  
  ➢ If so, then revise the guess.

- Continue for 2 hops, 3 hops, etc.
Dijkstra’s: Greedy algorithm

- Assume that every city is infinitely far away.

  - I.e., every city is $\infty$ miles away from BWI (except BWI, which is 0 miles away).

  - Now perform something similar to breadth-first search, and optimistically guess that we have found the best path to each city as we encounter it.

  - If we later discover we are wrong and find a better path to a particular city, then update the distance to that city.
Dijkstra’s algorithm

- Algorithm initialization:

  - Label each node with the distance $\infty$, except the start node, which is labeled with distance 0.
    - $D[v]$ is the distance label for $v$.
  
  - Put all nodes into a priority queue $Q$, using the distances as labels.
Dijkstra’s algorithm, cont’d

- While Q is not empty do:
  - \( u = Q.\text{removeMin} \)
  - for each node z one hop away from u do:
    - if \( D[u] + \text{miles}(u,z) < D[z] \) then
      - \( D[z] = D[u] + \text{miles}(u,z) \)
      - change key of z in Q to \( D[z] \)

- Note use of priority queue (Heap) allows “finished” nodes to be found quickly (in \( O(\log |V|) \) time).
An Example
Shortest mileage from BWI
Shortest mileage from BWI
Shortest mileage from BWI

- BOS: 371
- PVD: 328
- JFK: 184
- ORD: 621
- LAX: ∞
- DFW: 1575
- SFO: ∞
- MIA: 946
- BWI: 0
- 2342

Distances:
- 1464
- 802
- 846
- 1235
- 740
- 1391
- 1121
- 2704
- 867
- 184
- 1258
- 187
- 144
- 1090

Note: ∞ indicates an infinite distance.
Shortest mileage from BWI

- PVD: 328
- BOS: 371
- JFK: 184
- ORD: 621
- LAX: ∞
- DFW: 1575
- JFK: 184
- BWI: 0
- MIA: 946
- SFO: ∞
Shortest mileage from BWI

- SFO: 3075
- LAX: ∞
- DFW: 1575
- ORD: 621
- JFK: 184
- BWI: 0
- MIA: 946
- BOS: 371
- PVD: 328

Distances:

- SFO to LAX: 337
- SFO to DFW: 1464
- SFO to ORD: 1846
- SFO to JFK: 2704
- SFO to BOS: 867
- SFO to PVD: 187
- LAX to DFW: 1235
- LAX to JFK: 1391
- LAX to BOS: 144
- LAX to PVD: 1258
- DFW to JFK: 802
- DFW to BOS: 740
- DFW to PVD: 184
- ORD to JFK: 621
- ORD to BOS: 849
- ORD to PVD: 1090
- JFK to BOS: 946
- JFK to PVD: 1121
- BWI to MIA: 2342
- MIA to BOS: 1121
Shortest mileage from BWI

- SFO: 2467
- DFW: 1423
- ORD: 621
- JFK: 184
- BOS: 371
- PVD: 328
- MIA: 946
- LAX: ∞

Distances:
- SFO to DFW: 1464
- LAX to SFO: 337
- DFW to ORD: 802
- ORD to JFK: 740
- JFK to BOS: 187
- BOS to PVD: 144
- PVD to MIA: 1258
- MIA to LAX: 2342
- LAX to DFW: 1235
- DFW to JFK: 1391
- JFK to ORD: 621
- ORD to BWI: 184
- BWI to MIA: 1121
- MIA to JFK: 946
- JFK to LAX: 1090
- LAX to ORD: 187
- ORD to SFO: 2704
- SFO to JFK: 867
- JFK to BOS: 849
- BOS to PVD: 184
Shortest mileage from BWI

- SFO: 2467
- LAX: 3288
- DFW: 1423
- ORD: 621
- JFK: 184
- BOS: 371
- PVD: 328
- MIA: 946
- BWI: 0

Distances:
- SFO to LAX: 337
- LAX to DFW: 1235
- DFW to ORD: 1391
- ORD to JFK: 621
- JFK to BWI: 802
- BWI to MIA: 2342
- MIA to SFO: 1090
- SFO to ORD: 2704
- ORD to JFK: 867
- JFK to BOS: 849
- BOS to PVD: 144
- PVD to DFW: 1423
- DFW to BOS: 187
Shortest mileage from BWI

- SFO: 2467
- LAX: 2658
- DFW: 1423
- ORD: 621
- JFK: 184
- PVD: 328
- BOS: 371
- MIA: 946
- BWI: 0

Distances:
- SFO to LAX: 337
- LAX to DFW: 1235
- DFW to ORD: 1391
- ORD to JFK: 802
- JFK to PVD: 187
- PVD to BOS: 144
- BOS to MIA: 1090
- MIA to BWI: 2342
- BWI to SFO: 2704
- SFO to JFK: 1846
- JFK to ORD: 849
- ORD to BOS: 867
- BOS to DFW: 1423
- DFW to SFO: 2658
- SFO to JFK: 2467
- JFK to ORD: 1464
- ORD to JFK: 740
- JFK to BOS: 187
- BOS to MIA: 1258
- MIA to BWI: 1846
<table>
<thead>
<tr>
<th>City</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAX</td>
<td>2658</td>
</tr>
<tr>
<td>DFW</td>
<td>1423</td>
</tr>
<tr>
<td>ORD</td>
<td>621</td>
</tr>
<tr>
<td>JFK</td>
<td>184</td>
</tr>
<tr>
<td>SFO</td>
<td>2467</td>
</tr>
<tr>
<td>BOS</td>
<td>371</td>
</tr>
<tr>
<td>PVD</td>
<td>328</td>
</tr>
<tr>
<td>MIA</td>
<td>946</td>
</tr>
<tr>
<td>BWI</td>
<td>0</td>
</tr>
</tbody>
</table>

Shortest mileage from BWI
Shortest mileage from BWI

- SFO: 2467
- LAX: 2658
- DFW: 1423
- ORD: 621
- JFK: 184
- BOS: 371
- PVD: 328
- MIA: 946
- BWI: 0

Distances:
- SFO to LAX: 1235
- BOS to LAX: 1090
- JFK to LAX: 1464
- ORD to LAX: 740
- MIA to LAX: 2342
- DFW to LAX: 337
- BOS to LAX: 1258
- ORD to JFK: 1391
- JFK to ORD: 2704
- ORD to SFO: 1464
- JFK to SFO: 849
- LAX to SFO: 337
- DFW to JFK: 187
- JFK to DFW: 184
- ORD to JFK: 849
- ORD to DFW: 1391
- JFK to ORD: 2704
- ORD to LAX: 802
- LAX to JFK: 1121
- JFK to MIA: 946
- ORD to MIA: 1121
- MIA to JFK: 849
- BOS to MIA: 867
- BOS to JFK: 371
- JFK to BOS: 187
- ORD to BOS: 867
- LAX to BOS: 1846
- BOS to SFO: 867
- MIA to BWI: 2704
Classwork
Find the Shortest Paths from S
Dijkstra’s Algorithm is greedy

1. Optimization problem
   - Of the many feasible solutions, finds the minimum or maximum solution.

2. Can only proceed in stages
   - no direct solution available

3. Greedy-choice property:
   \textit{A locally optimal (greedy) choice will lead to a globally optimal solution.}

4. Optimal substructure:
   \textit{An optimal solution contains within it optimal solutions to subproblems}
Features of Dijkstra’s Algorithm

• “Visits” every vertex only once, when it becomes the vertex with minimal distance amongst those still in the priority queue

• Distances may be revised *multiple times*: current values represent ‘best guess’ based on our observations so far

• Once a vertex is finalized we are guaranteed to have found the shortest path to that vertex
Implementation
Heap is necessary to findMin

**Initialization**: $O(n)$

**Visitation loop**: $n$ calls

- `deleteMin()`: $O(\log n)$
- Each edge is considered only once during entire execution, for a total of $e$ updates of the priority queue, each $O(\log n)$

**Overall cost**: $O( (n+e) \log n )$
Question

- Dijkstra’s only finds the length of the shortest path
- Is it possible to modify the Dijkstra’s to actually find out the nodes in the shortest path?