Today’s Outline

• Announcements
  – Project 3 Code due Wed March 3 by 11pm
  – Written Homework #7 due Fri March 5
  – Project 3 Benchmarking & Written (and Above & Beyond) due Fri March 5 by 11pm

• Today’s Topics:
  – Graphs
    • Shortest Path Algorithms
      – Dijkstra’s Algorithm

Graphs so far

• Representations
• Topological Sort
• Finding paths
  – DFS
  – BFS
  – Dijkstra

Weighted SSSP: The Quest For Food

Can we calculate shortest distance to all nodes from Allen Center?

Dijkstra, Edsger Wybe

Legendary figure in computer science; was a professor at University of Texas.

Supported teaching introductory computer courses without computers (pencil and paper programming)

Supposedly wouldn’t (until very late in life) read his e-mail; so, his staff had to print out messages and put them in his box.

1972 Turning Award Winner.

Programming Languages, semaphores, and …

Dijkstra’s Algorithm: Idea

Adapt BFS to handle weighted graphs

Two kinds of vertices:
  – Finished or known vertices
  – Shortest distance has been computed
  – Unknown vertices
  – Have tentative distance
Dijkstra’s Algorithm: Idea

At each step:
1) Pick closest unknown vertex
2) Add it to known vertices
3) Update distances

Dijkstra’s Algorithm: Pseudocode

Initialize the cost of each node to $\infty$
Initialize the cost of the source to 0

While there are unknown nodes left in the graph
Select an unknown node $b$ with the lowest cost
Mark $b$ as known
For each node $a$ adjacent to $b$
   $a$’s cost = min($a$’s old cost, $b$’s cost + cost of ($b$, $a$))

Dijkstra’s Alg: Implementation

void Graph::dijkstra(Vertex s){
    Vertex v,w;
    Initialize s.dist = 0 and set dist of all other vertices to infinity
    while (there exist unknown vertices, find the one $b$ with the smallest distance)
        b.known = true;
        for each a adjacent to b
            if (!a.known)
                if (b.dist + Cost_ba < a.dist){
                    decrease(a.dist to= b.dist + Cost_ba);
                    a.path = b;
                }
    }
}

Activity

Weighted SSSP: The Quest For Food

If just wanted to know shortest path to Ben and Jerry’s could stop once Ben and Jerry’s is “known”

Dijkstra’s Alg: Implementation

Initialize the cost of each node to $\infty$
Initialize the cost of the source to 0

While there are unknown nodes left in the graph
Select the unknown node $b$ with the lowest cost
Mark $b$ as known
For each node $a$ adjacent to $b$
   $a$’s cost = min($a$’s old cost, $b$’s cost + cost of ($b$, $a$))
Dijkstra’s Algorithm: a Greedy Algorithm

Greedy algorithms always make choices that currently seem the best
- Short-sighted – no consideration of long-term or global issues
- Locally optimal - does not always mean globally optimal!!

Dijkstra’s Algorithm: Summary

- Classic algorithm for solving SSSP in weighted graphs without negative weights
- A greedy algorithm (irrevocably makes decisions without considering future consequences)
- Intuition for correctness:
  - shortest path from source vertex to itself is 0
  - cost of going to adjacent nodes is at most edge weights
  - cheapest of these must be shortest path to that node
  - update paths for new node and continue picking cheapest path

Correctness: The Cloud Proof

- Next shortest path from inside the known cloud
- Better path to B? No!
- How does Dijkstra’s decide which vertex to add to the Known set next??
  - If path to B is shortest, path to W must be at least as long
    (or else we would have picked W as the next vertex)
  - Only path through W to B cannot be any shorter!

Correctness: Inside the Cloud

Prove by induction on # of nodes in the cloud:
Initial cloud is just the source with shortest path 0
Assume: Everything inside the cloud has the correct shortest path
Inductive step: Only when we prove the shortest path to some node v (which is not in the cloud) is correct, we add it to the cloud

When does Dijkstra’s algorithm not work?

Dijkstra’s vs BFS

At each step:
1) Pick closest unknown vertex
2) Add it to finished vertices
3) Update distances

- Dijkstra’s Algorithm

At each step:
1) Pick vertex from queue
2) Add it to visited vertices
3) Update queue with neighbors

- Breadth-first Search

Some Similarities:

Activity

Negative-weight edges

- Why doesn’t Dijkstra’s work on graphs with negative-weight edges?
- Any ideas on how we could fix this?
The Trouble with Negative Weight Cycles

What’s the shortest path from A to E?

Analysis

• Total running time for Dijkstra’s:
  \[ O(|V|^2 + |E|) \] (linear scan)
  \[ O(|V| \log |V| + |E| \log |V|) \] (heaps)

What if we want to find the shortest path from each point to ALL other points?