CSE 326: Data Structures Dynamic Programming – Floyd/Warhsall Algorithm

Hal Perkins

Winter 2008

Lecture 26

Single-Source Shortest Path

- Given a graph **G** = (V, E) and a single distinguished vertex **s**, find the shortest weighted path from **s** to every other vertex in **G**.

All-Pairs Shortest Path:

- Find the shortest paths between all pairs of vertices in the graph.
- How?

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?

Dynamic Programming

- Algorithmic technique that systematically records the answers to sub-problems in a table and re-uses those recorded results (rather than re-computing them).
- **Simple Example**: Calculating the Nth Fibonacci number.

  Fib(N) = Fib(N-1) + Fib(N-2)

Dynamic Programming & Shortest Paths – Floyd-Warshall

- Given a directed graph G = (V, E) with no negative-weight cycles (negative-weight edges may be present), calculate the shortest paths between *all pairs* of vertices
- Idea: For each pair of verticies vi, vj, find shortest path from vi to vj that only passes through {v1, v2, …, vk }
  - Initially k=1. At each step, increase k by 1. Re-examine each pair vi, vj and see if using vk gives a shorter path than any discovered so far

Floyd-Warshall

- Data structure: M[x][y] contains the shortest known path from x to y. Initially this is just the adjacency matrix for the graph
- This version only shows the computation of the final path lengths – need additional bookkeeping to actually remember the paths

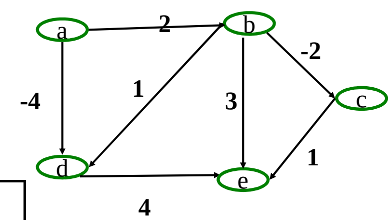
Floyd-Warshall

- Algorithm

```
for (int k = 1; k =< V; k++)
  for (int i = 1; i =< V; i++)
    for (int j = 1; j =< V; j++)
      if ( ( M[i][k]+ M[k][j] ) < M[i][j] )
        M[i][j] = M[i][k]+ M[k][j]
```

Invariant: After the kth iteration, the matrix includes the shortest paths for all pairs of vertices (i,j) containing only vertices 1..k as intermediate vertices

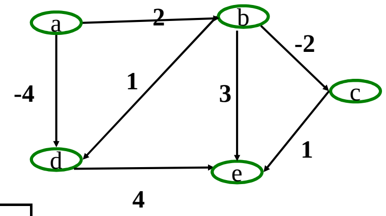
Initial state of the matrix:



	a	b	c	d	e
a	0	2	-	-4	-
b	-	0	-2	1	3
c	-	-	0	-	1
d	-	-	-	0	4
e	-	-	-	-	0

M[i][j] = min(M[i][j], M[i][k]+ M[k][j])

Floyd-Warshall - for All-pairs shortest path



	a	b	c	d	e
a	0	2	0	-4	0
b	-	0	-2	1	-1
c	-	-	0	-	1
d	-	-	-	0	4
e	-	-	-	-	0

Final Matrix Contents

9

Floyd-Warshall

- Algorithm

```
for (int k = 1; k =< V; k++)
  for (int i = 1; i =< V; i++)
    for (int j = 1; j =< V; j++)
      if ( ( M[i][k]+ M[k][j] ) < M[i][j] )
        M[i][j] = M[i][k]+ M[k][j]
```

- Total cost:
- Compared to running Dijkstra’s |V| times?

10