G205
Fundamentals of Computer Engineering

CLASS 19, Wed. Nov. 12 2003

Stefano Basagni

Fall 2003

M-W, 9:50am-11:30am, 410 Ell

Shortest Paths

◆ How to find the shortest route between two points in a map

◆ INPUT:
- A directed graph G=(V,E)
- A weight function w:E → **R**

◆ Weight of path p=<v_0, v_1,...,v_k > is
$$w(p)=SUM(i=1,k)w(v_{i-1}, v_i)$$

Weight of a Shortest Path

◆ Shortest-path weight from u to v

- $\delta(u,v)=\min\{w(p): p = u \rightsquigarrow v\}$
  if there is a path from u to v
- $\delta(u,v)=\infty$ otherwise

◆ Shortest-path from u to v

- Any path $p = u \rightsquigarrow v$ with weight

$$w(p) = \delta(u,v)$$

Single-Source Shortest Paths Output

- ◆ For v ∈ V the output is d[v]= ɗ(s,v)
  - Initially d[v]=∞
  - Reduces as algorithm progresses, but always d[v] ≥ ɗ(s,v)
  - Call d[v] a **shortest path estimate**
- ◆ π[v]=the predecessor of v in a path to s
  - If no predecessor, π[v]=NIL
  - π induces a tree—**Shortest-path tree**

Initialization

◆ All shortest-paths algorithms start with

Init-Single-Source(V,s)
  for each v ∈ V do
    d[v]=∞
    π[v]=NIL
    d[s]=0

Relaxation

◆ Can we improve the shortest-path estimated for v going through u and taking (u,v)?

Relax(u,v,w)
  if d[v]>d[u]+w(u,v)
    then   d[v]=d[u]+w(u,v)
            π[v]=u

Scheme for Single-Source Shortest-Paths Algorithms

◆ Start by calling Init-Single-Source

◆ Relax edges

◆ Different algorithms differ on
  - Number of relaxations
  - Order of relaxations

◆ Bellman-Ford

◆ Dijkstra

The Bellman-Ford Algorithm

◆ Allows negative-weight edges

◆ Computes d[v] and π[v] for each v ∈ V

◆ Returns **true** if no negative-weight cycle are reachable from s, **false** otherwise

The Algorithm

Bellman-Ford(V,E,w,s)
  Init-Single-Source(V,s)
  for i=1 to |V|-1 do
    for each edge (u,v)∈E do Relax(u,v,w)
  for each edge (u,v)∈E do
    if d[v]>d[u]+w(u,v) then return **false**
  return **true**

The Analysis

◆ Straightforward
  - Init-Single-Source takes Θ(V)
  - Relax takes constant time
  - First two nested for take O(VE)
  - Second for take O(V)

◆ Bellman-Ford takes O(VE) to produce all the shortest paths from a given source to all other nodes

BF Correctness,1

◈ No negative cycles

◈ Lemma: After the |V|-1 iterations of the first for of Bellman-Ford, for each v reachable from s is $d[v]=\delta(s,v)$.

Proof: Via the path-relaxation property.

Let $p=<v_0, v_1,...,v_k>$ be any acyclic shortest path from $s=v_0$ to $v=v_k$. Path p has at most |V|-1 edges: $k \leq |V|-1$.

BF Correctness, 2

Each of the $|V|-1$ iterations of the first for of Bellman-Ford relaxed all $|E|$ edges. Among the edges relaxed in the i-th iteration, $i=1,2,...,k$, (v_{i-1},v_i). By the path-relaxation property is then $d[v]=d[v_k]=\delta(s,v_k)=\delta(s,v)$.

BF Correctness, 3

◆ Corollary: For each $v \in V$ there is a path from s to v if and only if $d[v] < \infty$.

Proof: $\Rightarrow$ Previous lemma.

$\Leftarrow$ Let $d[v] < \infty$ and let us assume that v is not reachable from s. In this case is $\delta(s,v) = \infty$. But then $d[v] = \infty$ (no-path property) which contradicts $d[v] < \infty$.

BF Correctness, 4

◆ Theorem: Let Bellman-Ford run on a weighted, connected, directed graph G=(V,E) with weight function w:E → **R** and source s. If G contains no negative-weight cycles reachable from s, then the algorithm returns **true**, d[v]=ẟ(s,v) for all v ∈ V and the predecessor subgraph G_π is a shortest-paths tree rooted at s. If G contains a negative-weight cycle reachable from s then the algorithm returns **false**.

Shortest Paths on Directed Acyclic Graphs (DAGs)

◆ A DAG is a direct graphs with no cycles

◆ DAGs can be topologically sorted:

  - Linear ordering of DAG vertices so that is $(u,v) \in E$ then u appears before v in the ordering

◆ Application of DFS

DAG Topological Sort

Topological-Sort(G)

    call DFS(G) $\rightarrow$ finish times f[v]

    insert finished v at the front of a list

    return the linked list of vertices

◆Clearly $\Theta(V+E)$

Finding Shortest Paths

Dag-Shortest-Paths(G,w,s)

Topological-Sort(G)

Initialize-Single-Source(G,s)

for each u taken in topologically sorted order do

for each v $\in$ Adj[u] do

Relax(u,v,w)

DAG Shortest Paths, Analysis

◈ Topological-Sort is Θ(V+E)

◈ Initialize-Single-Source is Θ(V)

◈ Total of |E| iterations for the inner for

◈ Total time for finding d and π is thus

Θ(V+E)

◈ Linear in the size of the adjacency list representation of the graph

Assignments

◆ Textbook, Chapter 24, pages 588—595

◆ Updated information on the class web

page:

www.ece.neu.edu/courses/eceg205/2003fa