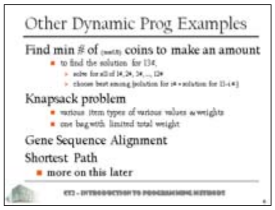
CS 2

Introduction to Programming Methods

Last Time

Dynamic programming

- how to solve optimization problems fast



Other Dynamic Prog Examples

Find min # of [illegible] coins to make an amount

- to find the solution for 13¢,
  - solve for all of 1¢, 2¢, 3¢, ..., 12¢
  - choose best among (solution for i¢ + solution for 13-i¢)

Knapsack problem

- various item types of various values & weights
- one bag with limited total weight

Gene Sequence Alignment

Shortest Path

- more on this later

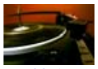
Networking

Communication between computers

- web, email, streaming, etc

Telephony

Video



Music

Social networks

Location

Mail

Cloud computing

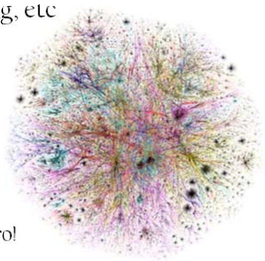
Networking

Communication between computers

- web, email, streaming, etc

Internet

- complex topology
  - computers connected
  - and routers
    - traffic directing



As usual, today is just an intro!

Addressing

Each machine has

- a globally unique address
  - IP address; 32 bits for now (www.caltech.edu: 50.18.115.211)
- and a domain name
  - easier for humans...
  - mapping from DN to IP may change dynamically
    - tree of DNS servers
      - » keep track of domains used recently

Addresses assigned by ICANN

- Marina del Rey

Circuit Switching

Used in (old) telephone network

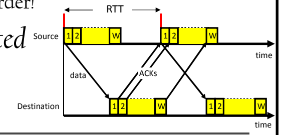
- when a call is made, switches are activated...
- establishing a line between two phones
  - quite rigid!
    - line busy even if no one is talking over it
    - if all lines are busy, you are stuck
  - but efficient
    - if you get a line, data can be sent reliably, with no overhead
- not quite appropriate for the internet...

Packet switching

Basic Principles

- data split into small (numbered) packets
- each packet is sent to receiver
  - so each can actually end up using a different path
- data reassembled at the receiver
  - can be received out of order!

Can be *connection-oriented* or *connectionless*



Connection-oriented Switching

"Virtual" Circuit Switching

- initial setup establishes route to destination
  - hops between network nodes; more on this later
- each node aware of "circuit"
  - stores connectionID to know where to forward packets
- then packets just include their connectionID
  - small header (minor overhead)
- "dedicated" line, in-order transmission
  - example: X.25, Frame-relay; also, TCP

Connectionless Switching

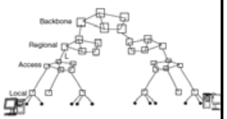
Datagram communication

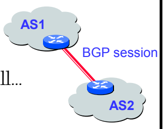
- each dispatched packet contains end address
- packets may go via different routes
  - based on target IP address
  - and which connection is active
    - routing tables dynamically updated
- example: Ethernet, IP, UDP
  - better for video conferencing or streaming
    - loss of a few packets not a showstopper

Heterogeneity of Networks

Two-level hierarchy

- Autonomous System (AS)
  - usually homogeneous domain
    - Sprint, AT&T, Verizon,...
  - shortest path routing: OSPF, IS-IS
- inter-domain
  - BGP (Border Gateway Protocol)
    - deal with inhomogeneity
  - the other protocols do not scale up well...
    - no guarantee of shortest path...

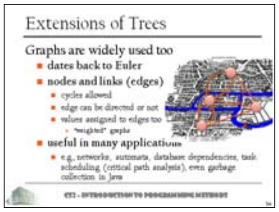




Abstraction

Networks are graphs

- extension of trees
- can be undirected
  - just a link btw nodes
- or directed
  - only in one direction
- and/or weighted
  - weigth can be, e.g., RTT

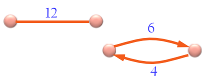


Extensions of Trees

Graphs are widely used too

- dates back to Euler
- nodes and links (edges)
  - cycles allowed
  - edge can be directed or not
  - values assigned to edges too
    - "weighted" graphs
- useful in many applications
  - e.g., networks, automata, database dependencies, task scheduling (critical path analysis), even garbage collection in Java

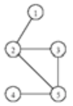
CS2 - Introduction to Programming Methods



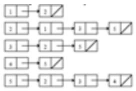
Storing Graph Connectivity

Two basic ways of encoding graphs

- adjacency matrix
  - M[i][j] indicates whether (i,j) is an edge
- linked list
  - or iterator
  - storing neighbors



$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$



Directed and/or weighted graphs?

- same deal, small changes

Single-Source Shortest Path

Find shortest paths from a source node

- used in OSPF (careful: only non-negative weights!)

Good news: dynamic programming at work

- if R is a node on the minimal path from P to Q, knowing the latter implies knowing the minimal path from P to R
- running time: O(E+V^2)
  - can be made O(E+V log V) with a priority queue

Pseudocode

Vertices stored in V

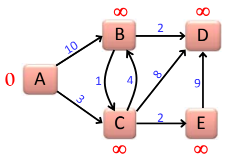
Source vertex = s

```
dist[s] ←0                                (distance to source vertex is zero)
for  all v ∈ V–{s}
     do  dist[v] ←∞                       (set all other distances to infinity)
Q←V                                       (queue Q initially contains all vertices)
while Q ≠Ø                                (while queue is not empty)
do   u ← mindistance(Q,dist)              (select/remove element of Q with min. distance)
     for all v ∈ neighbors[u]
          do  if  dist[v]>dist[u]+w(u, v)  (if new shortest path found)
               then d[v] ←d[u]+w(u, v)     (set new value of shortest path)
     remove u from Q
{if desired, add traceback code by udating, e.g., an array previous[v]}
return dist
```

Example I

Q: A B C D E

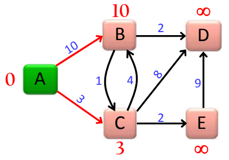
d: 0 ∞ ∞ ∞ ∞



Example II

Q: A B C D E

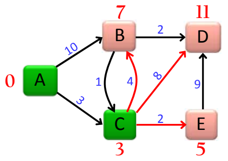
d: 0 10 3 ∞ ∞



Example III

Q: A B C D E

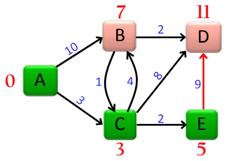
d: 0 7 3 11 5



Example IV

Q: A B C D E

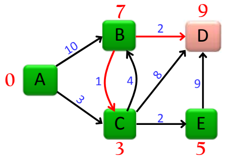
d: 0 7 3 11 5



Example V

Q: A B C D E

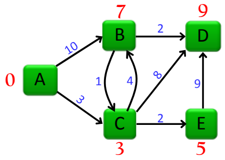
d: 0 7 3 9 5



Example VI – The End

Q: A B C D E

d: 0 7 3 9 5



Shortest Path in a Graph

Not just good for internet communication

- mapquest/Google map
  - roads are weighted by speed limit, average traffic, ...
- epidemiology
  - "contact" network, with weight equals probability
- robot motion planning
  - if you know a map of the environment

source: Wikipedia

[Edsger W. Dijkstra]

May 11, 1930 – August 6, 2002

- from theoretical physicist...
- to computer scientist
  - made a case against GOTO
  - made programming a science
    - Turing award in 1972

Graph Search

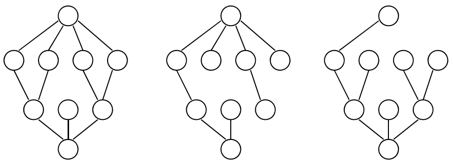
Two basic orders of search

- depth first search (DFS)
  - like trees' preorder: visit node, then its neighbors
- breadth first search (BFS)
  - same order as Dijkstra if equal weights
    - i.e., level by level
  - parallel version: MapReduce
    - programming model implemented in, e.g., Hadoop
    - basically, a tree of executions, then a gathering

Spanning Trees

Trees assembled during search

- BFS: short and bushy
- DFS: long and stringy



BFS from top

DFS from top

Other Typical Graph Problems

All-pair shortest paths

- dyn. prog. again, called Floyd-Warshall

$$\mathrm{shortestPath}(i,j,k) = \min(\mathrm{shortestPath}(i,j,k-1), \mathrm{shortestPath}(i,k,k-1)+\mathrm{shortestPath}(k,j,k-1))$$

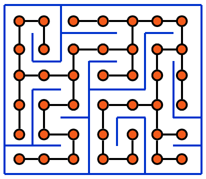
- if positive weights, Dijkstra's just as good
  - $O(V^3)$

Minimum spanning tree (e.g., power lines)

- find a subgraph of graph G
  - forming a tree containing all vertices of G
  - with min sum of weights
- Prim's algorithm (same complexity as SSSP)

[Spanning Trees]

Cool way to make a maze too…



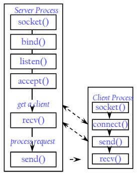
Final Words on Networking

Notion of ports

- different channels
  - IP header contains add + port

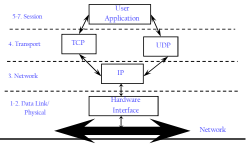
Notion of sockets

- chat over ports
  - listen/bind,recv/send,…



Notion of layers

DNS lookup	UDP	53
FTP	TCP	21
HTTP	TCP	80
POP3	TCP	110
Telnet	TCP	23


5-7. Session
User Application
4. Transport
TCP
UDP
3. Network
IP
1-2. Data Link/ Physical
Hardware Interface
Network