3134 Data Structures in Java

Lecture 17
Mar 28 2007
Shlomo Hershkop
Announcements

- Starting GRAPHS!!
Feedback

- Hw extensions...
- Hw itself
  - Where to begin...
- Serializable
  - What it is
  - How to use in your code
- Hashing in general in java
  - Where it is
  - How to use it
- testing
Today!

- We covered some nice data structures for manipulating groups of items and quickly finding a specific item

- Many times, we are also interested in relationships between groups of items (not just membership)

- Example: all people who called norway today 😊
Graph Theory!!
- very old, branch of math
- many open problems
- recent discoveries on interesting applications

Trees are linear DS since we have a root and leaves, so can start at specific point and end at a specific point

A more general DS would be to have 2 way links so can go anywhere

not clear where beginning is, so need to be careful about how algorithms will work
Definition

- A graph \( G=(V,E) \) consists of a set of vertices (nodes) \( V \) and edges \( E \) where each edge is a pair \( (v,w) \) such that \( v,w \) in \( V \)
- Directed Graph – A graph where vertices have direction

- Undirected Graph

- Weights – in addition can associate a weight with each edge
Example

- think of a map
- locations and mileage
- think of the internet
  - what are the nodes? edges?
Definition

- Path = a sequence of vertices $v_1, v_2, v_3...$ such that the vertices are adjacent

- Length = number of edges

- Loop = edges that starts and end at the same vertex
More definitions

- Simple path = a path with distinct vertices that doesn’t cross itself

- Cycle = path with the first and last vertex being the same
DAG ?
DAG

- Directed Acyclic Path
definitions

- connected graph = undirected graph where there is a path from any vertex to any other
Strongly Connected

- directed graph where path from any vertex to any other
more definitions

- weakly connected – directed graph which would have been connected if it was undirected

- Complete graph = every vertex connected to every other
Point

- Very general DS

- Air transit
  - airports
  - flights
  - weight = cost

- Can answer:
  - what are the cheapest flights?
  - shorted flights?
  - where we should add flights?
  - can we reach every city?
one last definition

- Degree of a node:
  - undirected
    - number of connections of the node
  - Directed
    - in degree
    - out degree
programming

- How would you code graphs?
some ideas:

- array

- linked list
Arrays

- 2 dimensional n by n array

- Let's do a simple example with 4 nodes
  - A, B, C, D

- Directed..

- How would an undirected graph be represented?
question

- how would we use weights here?

- how would the code look from a high level view?
Linked List

- you can also use a list representation called an adjacency list

- list of nodes
- linked list of edges
Resources

- how much space (in terms of V and E) would it take to represent a graph on either implementation?

- what does that tell you?
Searching

- graph can be used for many different things
- one operation we would like to support is searching the graph
- we want to visit nodes without wasting time or missing any
Example

- say you represent a maze as a graph

- Example...

- what strategy would you use to explore the maze??
Expansion

- DFS
  - depth first search

- BFS
  - breadth first search
Algorithm

- have a list of vertices and nodes
- choose one from the list
- put into DS X
- while X is not empty
  - choose y out of X
    - if not visited, visit
      - for each adj of y if not visited put into X

- what is X ??
- STACK
  - DFS
- Queue
  - BFS
We showed how to expand a graph to visit all nodes, any ideas on how to sort a graph?

What does it mean to sort a graph anyway?
Next week

- Do graph reading