Announcements

- Almost done...
Today

- Graphs Algorithms...
- Designing algorithms
- Beyond DS...

- Next class
  - Wrap up
- Next Monday
  - Review...
graph connectivity

- biconnected:
  - graph is biconnected is there are no vertices which can disconnect the graph
  - i.e no articulation points
    - points of failure
Simple fast solution

- Doing a DFS walk on a graph, we can find all articulation points
simple algorithm

- **DFS**
  - any vertex, DFS, number vertex as visited
  - check what is the lowest number you can reach from any edge and back edge
    - minimum of either:
      - current number
      - lowest out edge
      - lowest of any out edge
    - Need to keep both low and num for each vertex
  - Root is articulation only if more than one child
  - Anyone else: if Low >= Num
Algorithm design

- for solving any problem the algorithm is important
- but so is the data structure

- so how do you invent an algorithm??

- different classes of algorithms we have seen
Greedy Algorithm

- many examples that we have seen
- “grab what you can and run “ philosophy
- settle for an answer not quite best, hoping we get close
problem

- how would you program a cash register to spit out correct change for a transaction?

- let's say we want to minimize the number of coins we are returning
Approaches

- usually largest coin first

- so for 15 cents?

- runtime?

- what would happen if we have a 12 cent coin?
from the past

- huffman was an example of a greedy algorithm

- that is why we needed to keep the tree local to the encryption so we can decrypt the data
packing problem

say you are writing the software system for UPS/FEDEX how do you load the trucks going to destination x ??

given N items of size $s_1, s_2, ..s_n$

fewest number of bins to fit them in which each bin has a capacity

eexample capacity 1,

$.2 .5 .4 .7 .1 .3 .8$
constraints

- online vs offline

- real world online
  - what would be an algorithm
  - can you prove its optimal?
strategies

- next fit
  - fit in last or create new bin
  - runtime ?
  - we can prove that if optimal is M, it will not use more than 2M bins

- can you do the proof ??
proof

- consider 2 adjacent bins
- the sum must be greater than capacity (else they would be placed in one bin)
- which means we are wasting $< \frac{1}{2}$ the capacity
first fit

- scan all current bin and place in first bin which can hold the current item
  - runtime ?
  - can you improve it ?

- it wont use more than 17/10 M bins 😊
best fit

- find where it fits in best
  - runtime ?
  - worst case is exactly as first fit
  - easy to code
what can you say about the offline version of the problem?

can we get to optimal?

how?
divide and conquer

- we’ve seen many examples

- runtime typical?
Back to graphs....

Problem known as Euler circuits

Want to visit each edge, starting and ending at same node

- How can we figure out if the graph will let us
- Examples...
Graph needs to be connected
  - How can we figure this out?
Even degree nodes
  - Why?
Can also work if 2 nodes have odd degree, but then won’t start and end on same node
Related...

- Hamiltonian cycle
  - Can we traverse a graph by visiting each node exactly once.
  - Example: have unlimited travel budget and want to land at every airport exactly once starting and ending at same one.

- Any ideas on how to solve this?