CS W3134: Data Structures in Java
Lecture #6: Ordered lists, complexity, sort
9/23/04
Janak J Parekh

Administrivia

- Who has problems with command-line arguments on HW#1?
- Short demo of what to do with Song.java
- We might be switching a TA shortly; I'll keep you informed

Agenda

- Ordered lists
- Big-Oh notation (complexity)
- Sorting algorithms, if time allows
Ordered lists

- What’s an ordered list?
- How do we do…
  - Insert()? Book page 60 has a clever technique
    - Once you find the “right point”, slide down in a “bottom-up fashion”
  - Find()? Book page 57
- Binary search
  - Key: play the “number-guessing game”, but as an algorithm. Start in the middle and keep on cutting your search space by half. Let’s look at an example…

Costs

- How much do each of the previous operations cost in the worst case?
  - Most are linear, some are unit
- Binary search is special – it’s better than linear time
  - Divide the range by half until too small to divide further == # of comparisons needed
  - Reverse: what’s the range that can be covered with \( n \) steps? (Book page 63)
    - i.e., \( r = 2^{n} \)
- What’s this expressed as in terms of \( r \)?
  - \( s = \log_{2} r \)
- Algorithm grows logarithmically

Formalizing costs

- We’re going to approach this informally
- Time to insert one element is some constant \( K \)
  - e.g., \( T(1) = K \)
- Time to search for an element (linearly) is \( T(N) = K * N \)
- “Big-Oh Notation”: upper-bound on worst-case time
  - We drop the constant \( K \) – for sufficiently large \( N \), the constant is unimportant
  - To be precise, we find a function \( F(x) \), where \( T(x) \) is \( O(F(x)) \)
    - \( |T(x)| \leq K |F(x)| \) for some \( x > c \)
  - The idea of doubling your computer’s speed is embedded in \( K \)
  - \( T(N) = O(N) \), for example
Examples of costs

- For lists using arrays?
  - Linear search: $O(N)$
  - Etc.
  - Draw a graph of the comparative costs, page 72
- What are bad about arrays?
  - Slow search in unordered, slow insert in ordered – can we speed both? Yes
  - Fixed size: can we change that? Yes

Sorts

- Bubble (p. 85)
  - Sort pairwise repeatedly
  - Biggest placed each time
- Selection (p. 89)
  - Search for smallest, swap with first
  - Search for smallest, swap with second
- Insertion (p. 95)
  - Take the next one, and put it into the existing sorted subset
  - All $O(n^2)$
  - But they're not the exact same performance
  - Let's write out a little bit of psuedocode for each

Next Time

- Finish sorting
- Stacks