Quicksort and selection

Prof. Ramin Zabih

http://cs100r.cs.cornell.edu

Cornell University
Computer Science

Administrivia

- Assignment 2 is out, due in 2 pieces
  - Small amount is due this Friday
  - Most of it is due next Friday
- Quiz 1 and Assignment 1 are graded
  - You can check CMS
- Optional evening lecture(s) coming soon!
  - "Kindergarten cop" will be in a month or so
- Quiz 2 on Tuesday 9/18
  - Coverage through the next lecture

Recap from last time

- Big-O notation allows us to reason about speed without worrying about
  - Getting lucky on the input
  - Depending on our hardware
- Repeatedly removing the biggest element allows us to find the k^{th} largest
  - Problem is called "selection"
  - Algorithm is quadratic, $O(n^2)$

How to do selection better?

- If our input were sorted, we can do better
  - Given 100 numbers in increasing order, we can easily figure out the 5th biggest or smallest
- Very important principle!
  - Divide your problem into pieces
    - One person (or group) can provide `sort`
    - The other person can use `sort`
  - As long as both agree on what `sort` does, they can work independently
  - Can even "upgrade" to a faster `sort`
    - We will do this in CS100R

How to sort?

- Sorting is an ancient problem, by the standards of CS
  - First important "computer" sort used for 1890 census, by Hollerith
    - Strangely enough, we will talk about the census again shortly (the 1900 census)
- There are many algorithms
  - We will focus on quicksort

Quicksort summary

1. Pick an element (**pivot**)
2. **Partition** the array into elements <= pivot and > pivot
3. Quicksort these smaller arrays separately

- Example:

  [10 2 5 30 4 8 19]

- If the pivot is 8 we need to sort:

  [2 5 4 8] and [10 30 19]

  - Sort these using quicksort

The selection problem

- Rev. Charles L. Dodgson's problem
  - Based on how to run a tennis tournament
  - Specifically, how to award 2nd prize fairly

Quicksort and the pivot

- There are lots of ways to make quicksort fast, for example by swapping elements
  - We will cover these in section
- Notice that with a bad choice of pivot this algorithm does quite poorly
  - Suppose we happen to always pick the largest element of the array, or the smallest?
  - What does this remind you of?
- When can the bad case easily happen?

Modifying quicksort to select

- Suppose that we want to find the 5th largest element (4 are larger)
  - Quicksort kind of helps us do this already
  - When we partition the array, we know the element we want is in one smaller array!
  - So, we can figure out which smaller array we need to search
    - For example, if we have 10 elements, 7 <= pivot and 3 > pivot
    - We need to search for the 2nd largest element in the smaller array with 7 elements

Modified quicksort example

- Example: find 5th largest (< 4) in:

[8 3 4 6 3 8 5 10 10 7]

- The ones bigger than 8 are:

[10 10]

- So we need the 3rd largest (< 2) in:

[8 3 4 6 3 8 5 7]

Find element < k others in A

MODIFIED QUICKSORT:

- Pick an element in A as the pivot, call it x
- Divide A into A1 (<x), A2 (=x), A3 (>x)
- If k < length(A3)
  - Find the element < k others in A3
- If k > length(A2) + length(A3)
  - Find the correct element in A1
    - Let j = k – [length(A2) + length(A3)]
    - Find the element < j others in A1
- Otherwise, return x

What is the complexity of:

- Finding the 1st element?
  - O(1) (effort doesn't depend on input)
- Finding the biggest element?
  - O(n) (constant work per input element)
- Finding the median, by repeatedly finding and removing the biggest element?
  - $O(n^2)$ (linear work per input element)
- Finding the median using modified quicksort?
  - $O(n^2)$ (linear work per input element)

♦ What is the complexity of:

- Finding the 2nd biggest element (>all but 1)? The 3rd biggest (> all but 2)?
  - What do you think?
  - It's actually O(n)
  - We do 2 (or 3) "find biggest" operations
    - Each of which takes O(n) time
- Finding the element bigger than all but 5%?
  - Assume we do this by repeated "find biggest"
  - What if we use modified quicksort?