What You’ll Learn Today

– Review: selection sort and bubble sort
– On what basis should we compare algorithms?
– How else can we approach sorting?
  • A divide and conquer strategy
* A Mathematical Footnote

Actually, the running time is \((n^2-n)/2\), but as \(n\) becomes sufficiently large, the \(n^2\) part of this equation dominates the outcome. Hence the notation of \(O(n^2)\) algorithm.

A Different Approach to Sorting

Selection Sort and Bubble Sort have the same basic strategy:

- Find one item, and put it in place
- Repeat

How else might we approach this problem?

Hint: it takes fewer comparisons to sort a smaller list. How many comparisons does it take to sort 2 items?
Divide and Conquer Sorting

How would you sort these?

Bucket Sort Algorithm
– In a first pass, divide cards by suit
– In second pass, sort each suit

What is the running time?

Divide and Conquer Sorting

Quicksort uses a divide-and-conquer strategy. It is simpler and shorter to solve a smaller problem.

Basic strategy:
– Split the list based on a split value; put each item on one of 2 sub lists (great then/less than split value).
– Sort each sub list using the same approach
– Continue splitting and sorting sub lists until we get a list of length 1 (why?)
– Combine all of the sorted sub lists together to create the complete ordered list.
Quicksort Example

Begin with complete set of cards:

Split into two groups based on “less than 5” or “greater than 5”:

Repeat quicksort on each sublist:

Split into two groups based on “less than 3” or “greater than 3”:
Quicksort Example

Repeat quicksort on each sub list:

Until we have sub lists of length 1:

At the limit, we have this set of sub lists:

Finally, we combine into the complete, sorted list.
Quicksort Algorithm

```plaintext
Quicksort(list):
  if length of list > 1 then
    select splitVal
    for each item in list:
      if item < splitVal:
        add item to lesser
      if item > splitVal:
        add item to greater
    Quicksort(lesser)
    Quicksort(greater)
    list = lesser + splitVal + greater
```

Calculating the Running Time

How do we calculate the running time for Quicksort?

– Determine the number of comparisons.
– Determine the number of time we split the list.

Each time we want to split the list, we need to compare each item to the split value, and assign it to the correct sub-list.

– For a list of size n, we have n comparisons per split.
How many splits?

Running Time: Quicksort

How many times do we split a list of size \( n \)?

We keep splitting (in half) until we reach 1. How many splits is that?

- For \( n = 2 \), splits = 1
- For \( n = 4 \), splits = 2
- For \( n = 8 \), splits = 3
- For \( n = 16 \), splits = 4
- For \( n = 32 \), splits = 5
- For \( n = 64 \), splits = 6

What is the pattern here?
Running Time: Quicksort

Pattern: Each time we double the length of the list \( n \), we increase the number of splits by 1.

This is the opposite of the exponential relationship.
Recall that:
\[ 2^2 = 2 \times 2 = 4 \]
\[ 2^3 = 2 \times 2 \times 2 = 8 \]
\[ 2^4 = 2 \times 2 \times 2 \times 2 = 16 \]

Recall: Logarithms

The base-2 logarithm describes how many times we need to divide a value in half to obtain 1:

\[ \log_2(2) = 1 \]
\[ \log_2(4) = 2 \]
\[ \log_2(8) = 3 \]
\[ \log_2(16) = 4 \]
\[ \log_2(32) = 5 \]

\[ \log_2(n) = x \]
where \( x \) is the power to which we would raise 2 to obtain \( n \):
\[ 2^x = n \]
Running Time: Quicksort

Recall that for a list of size \( n \):
- We have \( n \) comparisons per split, and
- We have \( \log_2(n) \) splits.

Combining these, we can write
\[
\text{n times } \log_2(n) = n^* \log_2(n) \text{ steps}
\]
Quicksort is an \( O(n^* \log_2(n)) \) algorithm.

\( n^* \log_2(n) \) is always less than \( n^2 \).

Running Time Comparison
Sorting Algorithm Demo

A really cool graphical demo of different sorting algorithms running side-by-side:
http://www.cs.bu.edu/courses/cs101/demos sorts.html

(with thanks to Penny Ellard for the original page)

Also, check this out:
http://www.sorting-algorithms.com/

What You Learned Today

– Bucket sort
– Quick sort
– Running time analysis

– Which is better?
  • quicksort first and then binary search
  • Linear search without sorting first
Announcements and To Do List

- HW11 due TUE 11/23
- Readings:
  - http://www.sorting-algorithms.com/ (today)
  - http://www.hermit.cc/teach/HO/dbms/dbms.htm (Tuesday)
- **We will have class on Tuesday 11/23!**