Searching

11-24-2003

Opening Discussion

- What did we talk about last class?
- Do you have any questions about the assignment?
- How do you go about finding a certain card in a deck of cards? How do you find a name in a telephone book? What is the difference between the two and why does it matter?

Motivation

- So we have talked about using computers to store information and also about how we can reorder that information. However, it does us very little good unless we can find things in the data.
- Today we are going to talk about the process of looking for things in our data. The concept itself is very simple, but we want to do it right, not just get it done.

Sequential Searches

- The most basic method of searching for data is to perform an exhaustive, sequential search.
- This type of search goes one at a time and looks at every element to see if it is the one we are looking for. If it is, it returns it, otherwise it goes to the next one.
- This type of search is O(n). It is all we can do with unsorted data.

Advantages of Sorted Data

- Having sorted data gives us more power and flexibility in searching. This is because we now have information about the locations of things. We can look at one element and get some information about other elements automatically.
- With unsorted data we had to do exhaustive searches because each element was completely independent.

Breaking Data Up

- When data is sorted, each comparison gives us extra information. If we find one element with a lesser value and one with a greater value, we know that what we are looking for lies in between.
- Using this, we could do something simple like check 10 points at regular intervals and see which interval the data we wanted was in.

Binary Search

- Breaking data up once though isn't ideal. Instead what we want to do is break it up repeatedly. The simplest way of doing this is with a binary search.
- In this search we look at the middle element to see if it is greater or less than what we are looking for. This rules out half the data. We then look at the middle of whatever half it was in and repeat. This gives us O(log n) performance.

Code

- Let's now write code for doing a binary search. I have found that binary searches can be tricky if you aren't careful. Mainly, it is easy to have "off by one" errors that lead to infinite loops. To simplify the code I like to check at the start and end points in the beginning and make an invariant that the data is never at those locations.

Minute Essay

- Describe why the binary search gives us O(log n) performance. What are we counting when we say this?
- Next time we will look at advanced sorting.
- Assignment #7/8 is due today.