Algorithms: Linear and Binary Search

CS 110
Bryn Mawr College
Algorithm

• A well-defined set of instructions for solving a particular kind of problem.

• Algorithms exist for systematically solving many types of problems
  – Sorting
  – Searching
  – ...

Euclid's algorithm for greatest common divisor

• Problem:
  – Find the greatest common divisor of two numbers A and B
• GCD Algorithm
  1. While B is not zero, repeat the following:
     • If A > B, then A=A-B
     • Otherwise, B=B-A
  2. A is the GCD

```java
int A = 40902;
int B = 24140;

print("GCD of " + A + " and " + B + " is ");

while (B != 0) {
    if (A > B) {
        A = A - B;
    } else {
        B = B - A;
    }
}

println(A);
```
Exhaustive (Linear) Search

- Systematically enumerate all possible values and compare to value being sought
- For an array, iterate from the beginning to the end, and test each item in the array

Find "J"

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X |
Exhaustive (Linear) Search

// Search for a matching String val in the array vals. // If found, return index. If not found, return -1.

int eSearch(String val, String[] vals) {
    // Loop over all items in the array
    for (int i=0; i<vals.length; i++) {
        // Compare items
        int rslt = val.compareTo(vals[i]);

        if ( rslt == 0 ) { // Found it
            return i; // Return index
        }
    }
    return -1; // If we get this far, val was not found.
}
Binary Search

• Quickly find an item (val) in a sorted list.

• Procedure:
  1. Init min and max variables to lowest and highest index
  2. Repeat while min ≤ max
     a. Compare item at the middle index with that being sought (val)
     b. If item at middle equals val, return middle
     c. If val comes before middle, then reset max to middle-1
     d. If val comes after middle, reset min to middle+1
  3. If min > max, val not found

The most efficient way to play "guess the number" …
Binary Search

Find "J"
// Search for a matching val String in the String array vals
// If found, return index. If not found, return -1
// Use binary search.

int bSearch(String val, String[] vals) {
    int min = 0;
    int max = vals.length-1;
    int mid;
    int rslt;

    while (min <= max) {
        mid = int((max + min)/2); // Compute next index
        rslt = val.compareTo(vals[mid]); // Compare values
        if (rslt == 0) { // Found it
            return mid; // Return index
        } else if (rslt < 0) { // val is before vals[mid]
            max = mid - 1; // Reset max to item before mid
        } else { // val is after vals[mid]
            min = mid + 1; // Reset min to item after mid
        }
    }

    // If we get this far, val was not found.
    return -1;
}
An Experiment - Exhaustive vs. Binary Search

• For names (Strings) in arrays of increasing size...
  – Select 10 names at random from the list
  – Search for each name using Binary and Exhaustive Search
  – Count the number of iterations it takes to find each name
  – Plot number of iterations for each against list size

• Start with an array of 3830+ names (Strings)
Wow! That's fast!
Worst Case Running Time

• Exhaustive Search
  N items in a list
  **Worst case: Number of iterations = N**
  (we must look at every item)

• Binary Search
  After 1\textsuperscript{st} iteration, N/2 items remain (N/2\textsuperscript{1})
  After 2\textsuperscript{nd} iteration, N/4 items remain (N/2\textsuperscript{2})
  After 3\textsuperscript{rd} iteration, N/8 items remain (N/2\textsuperscript{3})
  ...
  Search stops when items to search (N/2\textsuperscript{K}) \rightarrow 1
  i.e. N = 2\textsuperscript{K}, \log_2(N) = K

  **Worst case: Number of iterations is log_2(N)**

*It is said that Binary Search is a logarithmic algorithm and executes in $O(\log N)$ time.*
List Size vs. Search Iterations
Binary vs. Exhaustive Search

\[ K = \log_2(N) \]
$K = \log_2(N)$

List Size vs. Search Iterations
Binary vs. Exhaustive Search
In Pictures: Weird Job Interview Questions

"Can I Guess?"
Given the numbers 1 to 1,000, what is the minimum number of guesses needed to find a specific number if you are given the hint "higher" or "lower" for each guess you make?
Asked at Facebook