Searching

Assumptions for search problems:

- Target is well defined.
- Target is fixed.
- Probes are accurate (hit or miss).
- Search domain is finite.
- We (can) remember all information gathered during search.

We search for a record with a **key**.

A Search Model

Problem:

Given:

- A list L, of n elements
- A search key X

Solve: Identify one element in L which has key value X, if any exist.

Model:

- The key values for elements in L are unique.
- Comparison determine $<$, $=$, $>$.
- Comparison is our only way to find ordering information.
- Every comparison costs the same.

Goal: Solve the problem using the minimum number of comparisons.

- Cost model: Number of comparisons.
- (Implication) Access to every item in L costs the same (array).

Is this a reasonable model and goal?

Linear Search

General algorithm strategy: Reduce the problem.

- Compare X to the first element.
- If not done, then solve the problem for $n-1$ elements.

```
Position linear_search(L, lower, upper, X) {
  if L[lower] = X then
    return lower;
  else if lower = upper then
    return -1;
  else return linear_search(L, lower+1, upper, X);
}
```

What equation represents the worst case cost?

Worst Cost Upper Bound

$$f(n) = \begin{cases} 1 & n = 1 \\ f(n-1) + 1 & n > 1 \end{cases}$$

Reasonable to guess that $f(n) = n$.

Prove by induction:

Basis step: $f(1) = 1$, so $f(n) = n$ when $n = 1$.

Induction hypothesis: For $k < n$, $f(k) = k$.

Induction step: From recurrence,

$$\begin{aligned} f(n) &= f(n-1) + 1 \\ &= (n-1) + 1 \\ &= n \end{aligned}$$

Thus, the worst case cost for n elements is linear.

Induction is great for verifying a hypothesis.

Approach #2

What if we couldn't guess a solution?

Try: Substitute and Guess.

- Iterate a few steps of the recurrence, and look for a summation.

$$\begin{aligned} f(n) &= f(n-1)+1 \\ &= \{f(n-2)+1\}+1 \\ &= \{\{f(n-3)+1\}+1\}+1\} \end{aligned}$$

Now what? Guess $f(n) = f(n-i)+i$.

When do we stop? When we reach a value for f that we know.

$f(n) = f(n-(n-1))+n-1 = f(1)+n-1 = n$

Now, go back and test the guess using induction.

Approach #3

Guess and Test: Guess the form of the solution, then solve the resulting equations.

Guess: $f(n)$ is linear.
  $f(n) = rn + s$ for some r, s.

What do we know?

- $f(1) = r(1) + s = r + s = 1$.
- $f(n) = r(n) + s = r(n-1) + s + 1$.

Solving these two simultaneous equations, $r = 1$, $s = 0$.

Final form of guess: $f(n) = n$.

Now, prove using induction.

Lower Bound on Problem

Theorem: Lower bound (in the worst case) for the problem is n comparisons.

Proof: By contradiction.

- Assume an algorithm A exists that requires only $n-1$ (or less) comparisons of X with elements of L.
- Since there are n elements of L, A must have avoided comparing X with $L[i]$ for some value i.
- We can feed the algorithm an input with X in position i.
- Such an input is legal in our model, so the algorithm is incorrect.

Is this proof correct?

Fixing the Proof

Error #1: An algorithm need not consistently skip position i.

Fix:

- On any given run of the algorithm, *some* element i gets skipped.
- It is possible that X is in position i at that time.

Error #2: Must allow comparisons between elements of L.

Fix:

- Include the ability to "preprocess" L.
- View L as initially consisting of n "pieces."
- A comparison can join two pieces (without involving X).
- The total of these comparisons is k.
- We must have at least $n - k$ pieces.
- A comparison of X against a piece can reject the whole piece.
- This requires $n - k$ comparisons.
- The total is still at least n comparisons.

Average Cost

How many comparisons does linear search do on average?

We must know the probability of occurrence for each possible input.

(Must X be in L?)

Ignore everything except the position of X in L. Why?

What are the $n + 1$ events?

$$\mathbf{P}(X \notin L) = 1 - \sum_{i=1}^{n} \mathbf{P}(X = L[i]).$$

Average Cost Equation

Let $k_i = i$ be the number of comparisons when $X = L[i]$.
Let $k_0 = n$ be the number of comparisons when $X \notin L$.

Let p_i be the probability that $X = L[i]$.
Let p_0 be the probability that $X \notin L[i]$ for any i.

$$
\begin{aligned}
f(n) &= k_0 p_0 + \sum_{i=1}^{n} k_i p_i \\
&= n p_0 + \sum_{i=1}^{n} i p_i
\end{aligned}
$$

What happens to the equation if we assume all p_i's are equal (except p_0)?

Computation

$$
\begin{aligned}
f(n) &= p_0 n + \sum_{i=1}^{n} ip \\
&= p_0 n + p \sum_{i=1}^{n} i \\
&= p_0 n + p \frac{n(n+1)}{2} \\
&= p_0 n + \frac{1-p_0}{n} \frac{n(n+1)}{2} \\
&= \frac{n+1+p_0(n-1)}{2}
\end{aligned}
$$

Depending on the value of p_0, $\frac{n+1}{2} \leq f(n) \leq n$.

Problems with Average Cost

- Average cost is usually harder to determine than worst cost.
- We really need also to know the variance around the average.
- Our computation is only as good as our knowledge (guess) on distribution.