CPS 100 - Program Design and Analysis II
Dr. Susan Rodger
Section: Analysis (handout)

Proof by Induction

$2^0 + 2^1 + 2^2 + 2^3 + \ldots + 2^N = ?$

Steps

1. Basis: Show simplest case is true (sometimes more than 1 case)
2. Induction Hypothesis (I.H.): Assume everything is true up to some cutoff point
3. Induction Step (I.S.): Show next step to be true. Must use I.H.

NOTE: ... is NOT a proof!

Proof of Problem above on N

1. Basis:

2. I.H.: Assume true for all ____
   $2^0 + 2^1 + 2^2 + \ldots +$

3. I.S.: Show true for ___
   $2^0 + 2^1 + 2^2 + \ldots +$

Analysis of Algorithms

- How fast?
- What do you measure?
  - no. of adds/mult
  - no. of loop executions
  - no. of comparisons
- Size of Input

Example 1

```
for (j=0; j<k; j++)
{
  sum = sum + A[j];
}
```

Example 2

```
for (j=0; j<N; j++)
  for (k=0; k<M; k++)
     cnt = cnt + A[j][k];
```

Big-Oh Notation

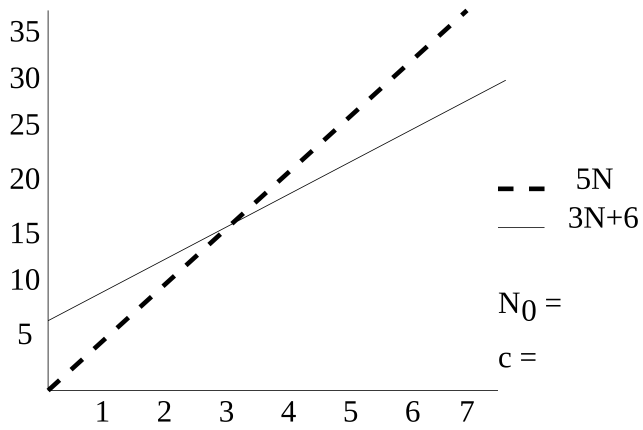
Definition: Algorithm A is order f(N), O(f(N)), if there exists constants c and N_0 such that A requires at most c∗f(N) steps to solve a problem of size $N \geq N_0$.

Example

Algorithm A takes: 3N+6

Claim: A is O(?)

Find c and N_0 such that $cN \geq 3N+6$ for all $N \geq N_0$



Example

Algorithm A takes 5N + 6 log N + $2N^2$

A is O(?)

Common Growth Rates

O(1)	constant
O(log N)	logarithmic
O(N)	linear
O(N^2)	quadratic
O(N^3)	cubic
O(2^N)	exponential

N	log N	N	N^2	N^3	2^N
4	2	4	16	64	16
8	3	8	64	512	256
16	4	16	256	4096	10^4
32	5	32	1024	10^4	10^9
64	6	64	4096	10^5	10^{19}
128	7	128	10^4	10^6	10^{38}
10^5	17	10^5	10^{10}	10^{15}	10^{30000}

Types of Analysis

Worst Case Analysis - maximum amount of time required

Average Case Analysis - average amount of time required

Example

Array of N integers in sorted order

- Find(X)
- Insert(X)
- FindMax()
- ListInSortedOrder()

Example

Array of N integers NOT in sorted order

- Find(X)
- Insert(X)
- FindMax()
- ListInSortedOrder()

Recursion

Factorial: $0! = 1$, $4! = 1 * 2 * 3 * 4$

```
int Fact(N)
{
   if (N ==0)
     return 1;
   else
     return N * Fact(N-1);
}
```

Analysis - Recursion

T(N) - time to solve a problem of size N

Analyze Fact

T(0) =

T(N) =

Recurrence Relation

Solve - want T() on one side of "="

T(N) =

Example: Silly Function

Don't interpret any meaning into this silly function!

```
int Biff(int n)
{
   int i, cnt=0;
   if (n=1) return 1;
   else
   {
     for (i=0; i<n; i++)
     {
       cnt++;
       if ((i==n/2) || i==(n-1))
          cnt = cnt+ Biff(n/2);
     }
   }
}
```

T(n) =

More on Linked Lists

Problem: Organize a linked structure of students in CPS 6 by Section number.

Pnode - Person Node

Snode - Section Node

see sectshort.cc