# Asymptotic Analysis

CSE 373
Data Structures & Algorithms
Ruth Anderson
Spring 2007

## Today's Outline

- **Admin**: Assignment #1 due next thurs. at 11:59pm
- **Asymptotic analysis**



# Asymptotic Analysis

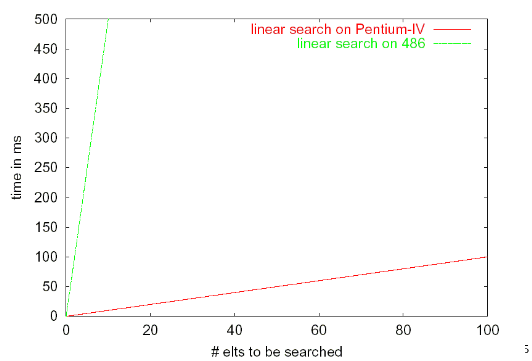
## Linear Search vs Binary Search

| | Linear Search | Binary Search |
|---|---|---|
| Best Case | | |
| Worst Case | | |

*So … which algorithm is better?*
*What tradeoffs can you make?*

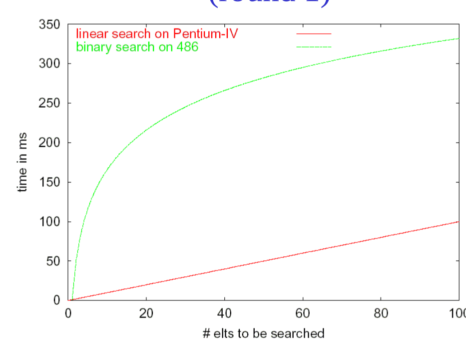


## Fast Computer vs. Slow Computer

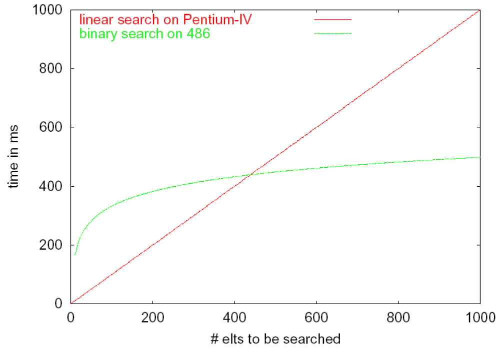





## Fast Computer vs. Smart Programmer (round 1)



# Fast Computer vs. Smart Programmer (round 2)




# Asymptotic Analysis

- Asymptotic analysis looks at the *order* of the running time of the algorithm
  - A valuable tool when the input gets "large"
  - Ignores the *effects of different machines* or *different implementations* of the same algorithm
- Intuitively, to find the asymptotic runtime, throw away the constants and low-order terms
  - Linear search is $T(n) = 3n + 2 \in \mathbf{O(n)}$
  - Binary search is $T(n) = 4 \log_2 n + 4 \in \mathbf{O(\log n)}$

*Remember: the fastest algorithm has the slowest growing function for its runtime*



# Asymptotic Analysis

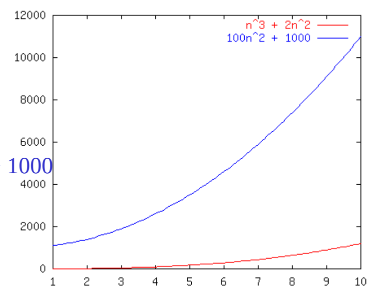
- Eliminate low order terms
  - $4n + 5 \Rightarrow$
  - $0.5\, n \log n + 2n + 7 \Rightarrow$
  - $n^3 + 2^n + 3n \Rightarrow$
- Eliminate coefficients
  - $4n \Rightarrow$
  - $0.5\, n \log n \Rightarrow$
  - $n \log n^2$ =>



# Order Notation: Intuition

$f(n) = n^3 + 2n^2$

$g(n) = 100n^2 + 1000$




Although not yet apparent, as $n$ gets "sufficiently large", $f(n)$ will be "greater than or equal to" $g(n)$



# Definition of Order Notation

- Upper bound: $T(n) = O(f(n))$ Big-O

  Exist constants $c$ and $n'$ such that

  $T(n) \le c\, f(n)$ for all $n \ge n'$
- Lower bound: $T(n) = \Omega(g(n))$ Omega

  Exist constants $c$ and $n'$ such that

  $T(n) \ge c\, g(n)$ for all $n \ge n'$
- Tight bound: $T(n) = \theta(f(n))$ Theta

  When both hold:

  $T(n) = O(f(n))$

  $T(n) = \Omega(f(n))$



# Order Notation: Definition

**O( f(*n*) )** : <u>a set or class of functions</u>

$g(n) \in O(f(n))$ iff there exist consts $c$ and $n_0$ such that:

$g(n) \le c\, f(n)$ for all $n \ge n_0$

Example: $g(n) = 1000n$ vs. $f(n) = n^2$

Is $g(n) \in O(f(n))$ ?

Pick: n0 = 1000, c = 1

## Notation Notes

**Note**: Sometimes, you'll see the notation:

$$g(n) = O(f(n)).$$

This is equivalent to:

$$g(n) \in O(f(n)).$$

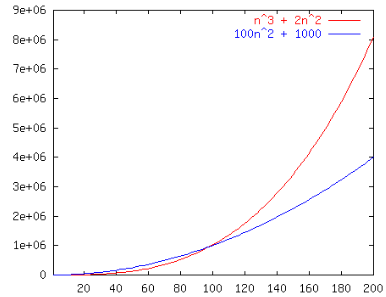
**However**: The notation

$O(f(n)) = g(n)$ is meaningless!

(in other words big-O is not symmetric)



## Order Notation: Example




$100n^2 + 1000 \le 5 (n^3 + 2n^2)$ for all $n \ge 19$

So $f(n) \in O(g(n))$



## Big-O: Common Names

| | | |
|---|---|---|
| – constant: | $O(1)$ | |
| – logarithmic: | $O(\log n)$ | ($\log_k n$, $\log n^2 \in O(\log n)$) |
| – linear: | $O(n)$ | |
| – log-linear: | $O(n \log n)$ | |
| – quadratic: | $O(n^2)$ | |
| – cubic: | $O(n^3)$ | |
| – polynomial: | $O(n^k)$ | (k is a constant) |
| – exponential: | $O(c^n)$ | (c is a constant > 1) |



## Meet the Family

- $O(f(n))$ is the set of all functions asymptotically less than or equal to $f(n)$
  - $o(f(n))$ is the set of all functions asymptotically strictly less than $f(n)$
- $\Omega(f(n))$ is the set of all functions asymptotically greater than or equal to $f(n)$
  - $\omega(f(n))$ is the set of all functions asymptotically strictly greater than $f(n)$
- $\theta(f(n))$ is the set of all functions asymptotically equal to $f(n)$



## Meet the Family, Formally

- $g(n) \in O(f(n))$ iff
  There exist $c$ and $n_0$ such that $g(n) \le c\, f(n)$ for all $n \ge n_0$
  - $g(n) \in o(f(n))$ iff
    There exists a $n_0$ such that $g(n) < c\, f(n)$ for all $c$ and $n \ge n_0$
    Equivalent to: $\lim_{n\to\infty} g(n)/f(n) = 0$
- $g(n) \in \Omega(f(n))$ iff
  There exist $c$ and $n_0$ such that $g(n) \ge c\, f(n)$ for all $n \ge n_0$
  - $g(n) \in \omega(f(n))$ iff
    There exists a $n_0$ such that $g(n) > c\, f(n)$ for all $c$ and $n \ge n_0$
    Equivalent to: $\lim_{n\to\infty} g(n)/f(n) = \infty$
- $g(n) \in \theta(f(n))$ iff
  $g(n) \in O(f(n))$ and $g(n) \in \Omega(f(n))$



## Big-Omega et al. Intuitively

| Asymptotic Notation | Mathematics Relation |
|---|---|
| $O$ | $\le$ |
| $\Omega$ | $\ge$ |
| $\theta$ | $=$ |
| $o$ | $<$ |
| $\omega$ | $>$ |

# Pros and Cons of Asymptotic Analysis

04/04/08 19

# Types of Analysis

Two orthogonal axes:

- bound flavor
  - upper bound (O, o)
  - lower bound (Ω, ω)
  - asymptotically tight (θ)
- analysis case
  - worst case (adversary)
  - average case
  - best case
  - “amortized”

04/04/08 20

# Algorithm Analysis Examples

- Consider the following program segment:

```
x:= 0;
for i = 1 to N do
  for j = 1 to i do
    x := x + 1;
```

- What is the value of x at the end?

04/04/08 21

# Analyzing the Loop

- Total number of times x is incremented is executed =

$$1+2+3+\ldots=\sum_{i=1}^{N} i=\frac{N(N+1)}{2}$$

- Congratulations - You’ve just analyzed your first program!
  - Running time of the program is proportional to N(N+1)/2 for all N
  - Big-O ??

04/04/08 22

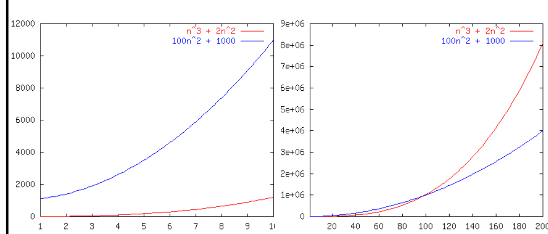
# Which Function Grows Faster?

$n^3 + 2n^2$ vs. $100n^2 + 1000$

04/04/08 23

# Which Function Grows Faster?

$n^3 + 2n^2$ vs. $100n^2 + 1000$




04/04/08 24