Lecture 10

Solving second-degree, homogeneous linear r. r. with constant coefficients (SOHLrrCC) Con't and Time Efficiency of Algorithms

Summary so far:

When the characteristic equation of a SOLHCC rr has **2 distinct roots**, r and s, we know that the sequences $r^0, r^1, r^2, \ldots$ and $s^0, s^1, s^2, \ldots$ satisfy the original SOLH rr CC (Lemma 8.3.1).

Lemma 8.3.2 assures us that if $r_0, r_1, r_3, \ldots$ and $s_0, s_1, s_2, \ldots$ are 2 sequences that satisfy a given SOLH rr, then sequence $a_0, a_1, a_2, \ldots$ where $a_n = C\, r_n + D\, s_n$ *also satisfies the recurrence relation.*

Theorem 8.3.3 proves that *EVERY sequence* that satisfies the original SOLHrrCC is of this form, i.e., has the form $C\, r_n + D\, s_n$.

So to find the "formula" for all sequences that satisfy the recurrence:

What if the characteristic equation has only 1 root, r?

Let the root be r. Then $r^0, r^1, r^2, \ldots$ is one sequence, and by definition

$0, r^1, 2r^2, 3r^3, 4r^4, \ldots, nr^n, \ldots$ is also a sequence (by Lemma 8.3.4)

Just as in the 2 roots case, we can write the sequence that satisfies the SOLHrrCC with a single expression—slightly different from the 2-roots expression:

$$a_n = C \cdot r^n + D \cdot \boldsymbol{n} \cdot r^n \qquad \text{(Theorem 8.3.5: The Single Root Theorem)}$$

As in the 2-roots case, use the initial conditions to find

---

ex. $a_k = 2a_{k-1} - a_{k-2}$ $\quad a_0 = 1, \quad a_1 = 2$

The root r =

The single sequences has the form:

Use the initial conditions, i.e. a_0 and a_1:

ex. $a_k = 6\, a_{k-1} - 9\, a_{k-2}$ $a_0 = 2$ $a_1 = 3$

ex. $a_k = 8\, a_{k-1} - 16\, a_{k-1}$ $a_0 = 2$ $a_1 = 4$

Summary of solving second-degree, linear homogeneous recurrence relations:

1. Write and solve the characteristic equation.

2. Write 2 sequences r_i and s_i that satisfy the r. r.
   Case 1: There are 2 roots of the char. equation, r and s.

   Case 2: There is only 1 root, r.

3. Since all solutions are of the form $C\, r_i + D\, s_i$ (2 roots) or $C\, r_i + D \cdot n \cdot r_i$ (1 root) use the initial conditions to find C and D.

4. Write $a_n = C\, r^n + D\, s^n$ *or* $a_n = C\, r^n + D \cdot n \cdot r^n$ **(This is closed form!)**

Time Efficiency of Algorithms

```
for (int  i = 0; i < N; i ++ )
  System.out.println (A[i]);
```

T i m e

Number of Items (N) →

What is this algorithm doing to array A?

```
for (int  i = 0; i < N; i ++ )
    for ( int j = 0; j < N - i - 1; j ++)
        if ( A[j] > A[j+1])
            swap ( A[j], A[j+1]);
```

T i m e

Number of Items (N) →

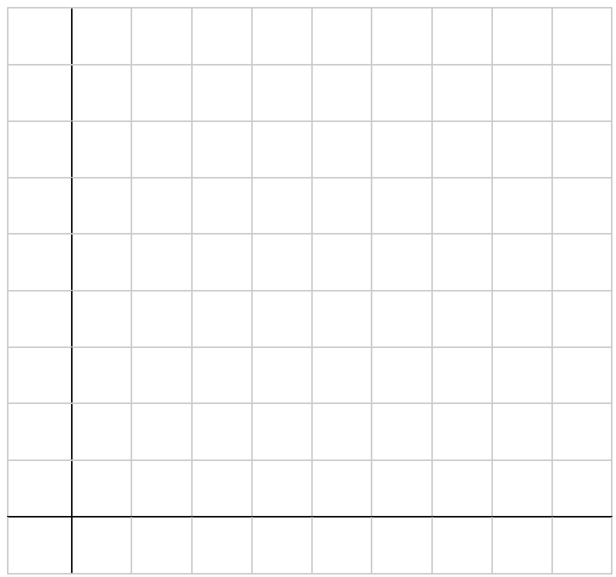
Aim: Place algorithms in categories that exhibit similar growth rates as N increases.

Graphing Functions

Real-valued functions, domain is **R** or domain is **Z**.

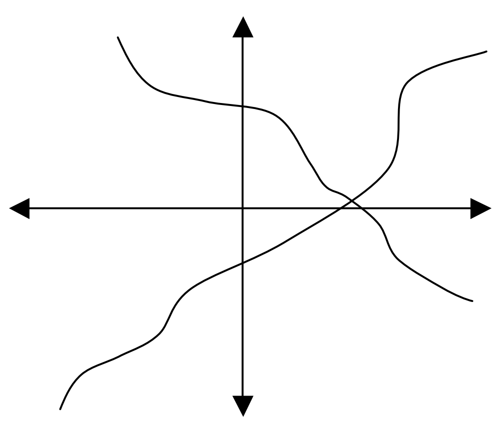
Power functions: $y = x^a$ written as $p_a(x) = x^a$

ex. $y = p_1(x) =$ $y = p_{1/2}(x) =$ $y = p_2(x) =$



Function f is **increasing** on a set S of its domain, if $\forall$ x_1 and $x_2 \in S$ where $x_1 < x_2$, $f(x_1) < f(x_2)$.

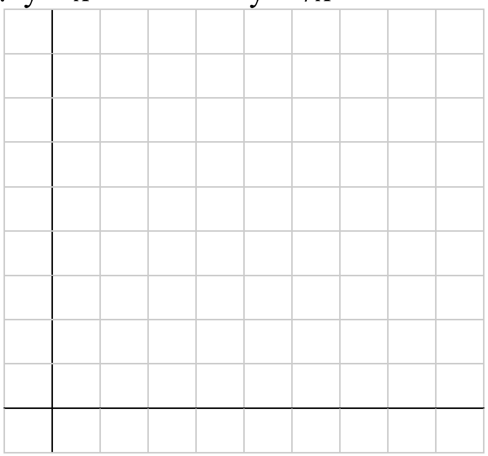
Function f is **decreasing** on a set S of its domain,



What effect does multiplying a function by a positive constant have on the shape of the graph?

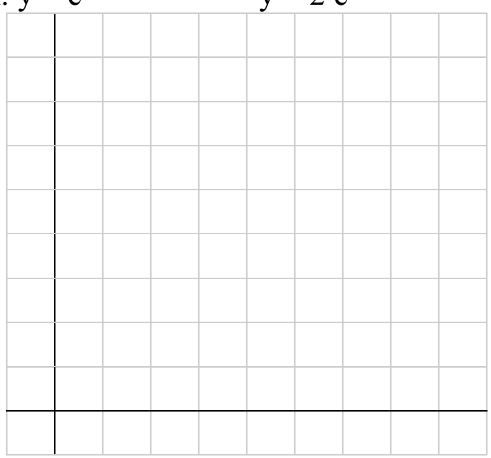
$y = f(x)$ multiply by c → $y = c\,f(x)$

ex. $y = x$ $y = 7x$

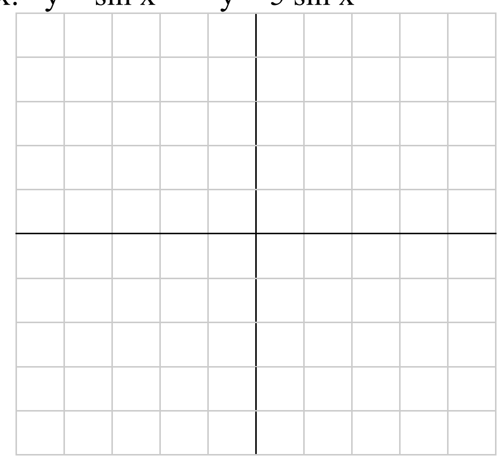


ex. $y = 3x + 1$ $y = 3\,(3x + 1)$

ex. $y = e^x$ $y = 2\,e^x$



ex: $y = \sin x$ $y = 5 \sin x$



"stretching factor" (from the x-axis)

Big-O

Def: We will say that f(x) is **O(g(x))** if there exists an x value $\mathbf{x_0}$ and some constant **M** such that

$$| f(x) | \leq \mathbf{M} \cdot | g(x) | \text{ for all } x > x_0.$$

Informally, we say that f(x) is in the O(g(x)) "family"

Note: Since in CS we are concerned with Time and time (f(x)) is always nonnegative, the inequality is

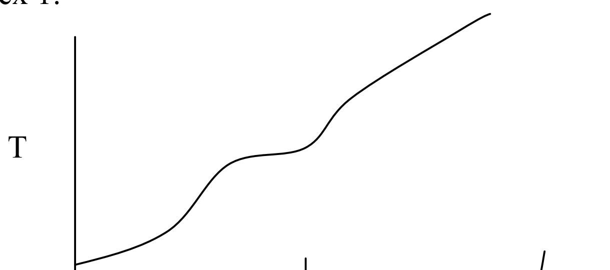
$$f(x) \leq M \cdot g(x) \quad \text{for all } x > x_0.$$

M is some "stretching factor" for g(x).

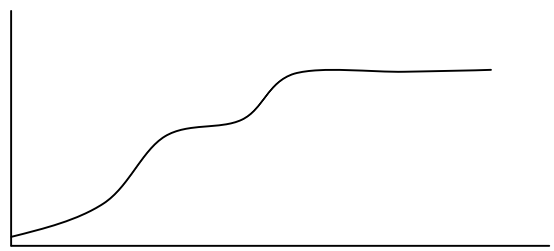
So we can "stretch" g(x) and find that the stretched version is an upper bound on ______________ for all x values that are _______________________.

Choose "simple" functions for g(x).

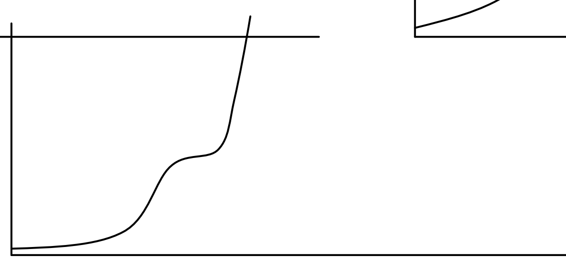
ex 1:



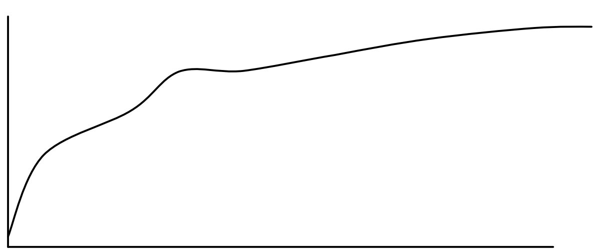
ex 2:



ex 3:



ex 4:



??? Is the choice of the g(x) for O(g(x)) unique?

Proving a Function is O(g(x))

Recall the definition of big-O:

f(x) is **O(g(x))** if there exists an x value $\mathbf{x_0}$ and some constant **M** such that
$| f(x) | \leq \mathbf{M} \cdot | g(x) |$ for all $x > x_0$.

Prove $f(x) = 3x^2 + 5x + 8$ is $O(x^2)$

We need a "stretching" factor, M and a x_0 so that
$| f(x) | \leq M \cdot | g(x) |$ for all $x > x_0$.

$| f(x) | = |3x^2 + 5x + 8|$

1. Use Δ Inequality to "break up" the terms
2. Remove | | and list conditions.
3. Substitute lower powers with the higher powers.
   List conditions. ($x > a$, $x > b$, ….)
4. Combine terms to find M
5. Choose the largest value (steps 2 & 3) as x_0

Is f(x) also $O(x^3)$?

$O(x^4)$?

$O(x)$?

ex. Show $f(x) = 3x^5 + 5x^3 - 4x$ is $O(x^5)$.

Polynomial functions are $O(x^h)$ where h is the highest power in the polynomial. **(The highest power term "dominates" all other terms.)**

ex. Show $f(x) = \mid 2x^2 - 5x - 3 \mid$ is $O(x^2)$.

ex. Show $f(x) = 5 \cdot 2^x + 3x^4$ is $O(2^x)$.

*Big-O can be a "loose" bound. We would like the **best** Big-O—answer THETA.*

Positive function f(x) is $\Theta(g(x))$ if there exists an x value x_0 and some constants M and N such that $N \cdot g(x) \le f(x) \le M \cdot g(x)$ for all $x > x_0$.

Intuition:

BTW, there is a "partner to big-O. It is Ω. A function f is $\Omega(g(x))$ if $\exists$ M and $x_0 \ni$ $M \cdot g(x) \le f(x) \quad \forall \; x > x_0$.

Intuition:

So if f(x) is both O(g(x) and Ω(g(x)), then it is

<u>To prove f(x) is Θ (g(x)),</u> show

1. f(x) is O(g(x)) (i.e., find "stretching" factor M and x_0 such that $f(x) \leq M \cdot g(x)$ for all $x > x_0$.)
2. Find the lower "stretching" factor N and x_0 such that $N \cdot g(x) \leq f(x)$ for all $x > x_0$. (i.e. show that f(x) is _____________)
3. Choose the larger of the 2 x_0 as the final x_0. Rewrite the triple inequality, replacing N and M.

Show $f(x) = 7x + 3$ is $\Theta(x)$

Step 1: (big-O)

Step 2:

1. Remove | | and list the conditions.
2. Make quantity smaller by substituting or omitting terms. List conditions.
3. Combine terms to find N
4. Choose the largest value (steps 1 & 2) as x_0

Step 3:

Show $f(x) = 2x^2 - 3x$ is $\Theta(x^2)$.

Show $f(x) = | 3x^2 - 5x + 4|$ is $O(x^2)$.

Show $f(x) = 3 \cdot 2^x + 3 x^3$ is $\Theta(2^x)$.

When a computer scientist says an algorithm is big-O of something, he/she usually means it is Θ of something (tight big-O).