Algorithms HW #2

1.) Order & Explain Asymptotic Growth

a.) $1000n \log n < n^2$

1 - Remove Constant factors

$\cancel{n} \log n < n^{\cancel{2}} \Rightarrow \log n < n \quad \left(\frac{n}{\log n}\right)' = \frac{1}{0}$

The logarithm of n grows considerably slower than n.

b.) $\sqrt{\cancel{10n^2} + 200n^3} \quad \sqrt[3]{\cancel{n^4} + \cancel{120n} + 30n^5 \cancel{+100}}$

2 - Only Highest Term Matters

1 $\cancel{200}n^{3/2}$?? $\cancel{30}n^{5/3} \Rightarrow n^{3/2} < n^{5/3}$

c.) $n!$ & n^n

$n! = n(n-1)(n-2)\cdots 1$
$n^n = n(n)(n)\cdots(n)$

Can clearly see each term in n^n is larger.

Alternatively:

Sterlings Approximation

$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$

$$\frac{n^n}{n!} = \frac{n^n}{\sqrt{2\pi n}\,\frac{n^n}{e^n}} = \frac{\cancel{n^n} e^n}{\cancel{n^n}\sqrt{2\pi n}} = \left(\frac{e^n}{\sqrt{2\pi n}}\right)' \Rightarrow \infty$$

So, n^n grows asymptotically faster than $n!$

d.) n^2 & $n \cdot 2^n$

Largest Factor & LHP

$\frac{2^n}{n^2} \Rightarrow \frac{n2^{n-1}}{2n} \Rightarrow \frac{(5)2^{n-n}}{2}$

$\Rightarrow \infty \therefore 2^n > n$

e.) n^{n^2} & 2^{2^n}

Take the log: $n^2 \log n$?? $2^n \cancel{\log 2}$ 1

$n^2 \log n < 2^n \quad \frac{2^n}{n^2 \log n}$ LHP

$\Rightarrow \infty$

2.) Prove or Provide Counterexample:

a.) if $f, g \in O(h)$, then $fg \in O(h)$

Counterexample

$h = 7n \quad O(h) = n$

$f = 3n \quad f \in O(h)$
$g = 2n \quad g \in O(h)$

$fg = 6n^2 \quad fg \notin O(h) \;/\; n^2 >> n$

b.) If $f \in O(g)$, then $f^n \in O(g^n)$ for every positive n.

True

$f \in O(g) \Rightarrow f \leq g$

f^n ?? $g^n \quad \cancel{n} \log f$?? $\cancel{n} \log g$

Some constant

Case 1

- $\log(f) \quad \log(g)$ LHP $\Rightarrow \frac{\log f}{\log g} \Rightarrow \frac{g}{f} = c$

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

Case 2

$\log(f) \quad \log(g)$ LHP $\Rightarrow \frac{\log f}{\log g} \Rightarrow \frac{g}{f} = \infty$

$\therefore O(f^n) \in O(g^n)$

c.) If $f \in O(g)$, then $2^f \in O(2^g)$

True.

Pf Let $2^f \notin O(2^g)$, suppose not,

$O(f)$ would need to be $> O(g)$

for 2^f to grow asymptotically faster than 2^g.

but we know $f \in O(g)$

$\therefore O(f)$ can't be $> O(g) \therefore$ we have a contradiction, 2^f must $\in O(2^g)$.

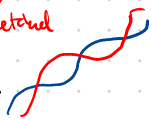
Two Monotonically increasing functions f, g s.t.

$f \notin O(g)$ $g \notin O(f)$

$$f = \begin{cases} 2x + \sin x & \text{if } n \text{ is odd} \\ 2x + \cos x & \text{if } n \text{ is even} \end{cases}$$

$$g = \begin{cases} 2x + \cos x & \text{if } n \text{ is odd} \\ 2x + \sin x & \text{if } n \text{ is even} \end{cases}$$

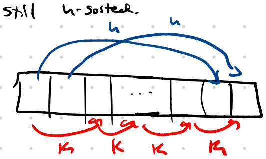
I had this construction made and intuitively constructed the double helix graph but didn't know how to construct it, tried $2^n/2^{n-1}$, considered recursion but could not think of how to implement, found a stack exchange thread that mentioned $2x + (\cos/\sin)$ which modeled the behavior I originally sketched



cs.stackexchange/questions/10548

4) $\forall\, h, k > 0$, if h-sorted array is k-sorted & is still h-sorted.

idea



K is some constant multiple of h and a shift, so h will have already resolved any out of order #s K would be able to displace.

H-Sorted $A[i] \le A[i+h]$ K-Sorted $A[i] \le A[i+k]$

After H-Sorting our array has the property

$A[i] \le A[i+h]$, then we K sort.

$A[i] \le A[i+k]$, $A[i+k] \le A[i+2k]$,

$A[i] \le A[i+k] \le A[i+2k] \le \cdots \le A[i+mk]$

h can be written as $mk + c$ where $c < k$

Since A is h-sorted & $i + h < n \rightarrow A[i] \le A[i+h]$