NANYANG TECHNOLOGICAL UNIVERSITY

SC2001/ CX2101: Algorithm Design and Analysis

Week 8: Review Lecture

Huang Shell Ying

Revision of Complexity Analysis

- Complexity analysis expressed in big-O, big-θ, big-Ω gives the <u>growth rate</u> of a function compared with another function – the order of magnitude of increase in the function when n increases. E.g $f(n) = O(n^2)$, $g(n) = O(n)$
- Two functions with the same complexity class may have very different running times. E.g $f(n) = 2n^2$, and $g(n) = 10n^2$

Revision of Complexity Analysis

- Arrange the following functions in increasing order of their big-O time complexity

  $f(n)= n^2$, $\log n$, $n \lg n$, $n!$, 2^n, 3^n, $\lg n$, n, $n^{1/2}$, n^5, $\log(n^2)$, 2^{2n}, 1000, n^n

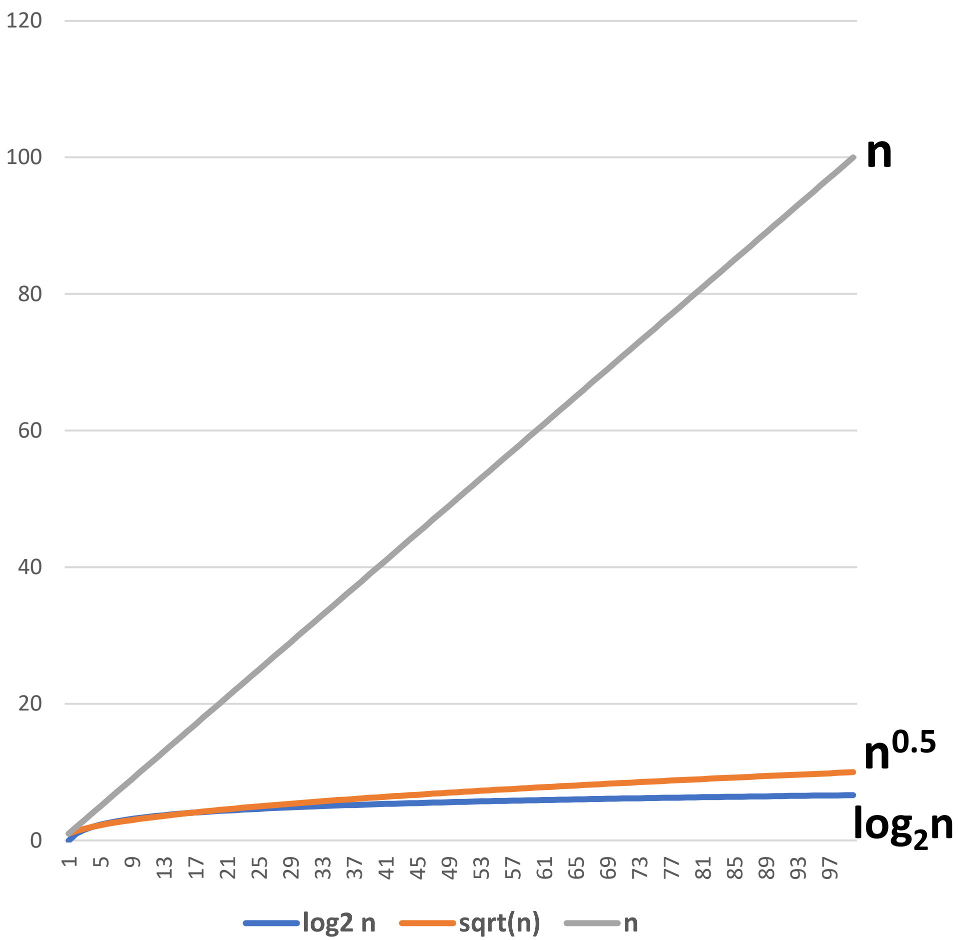
$\log_{10}(x) = \ln(x) / \ln(10)$
$\log_{10}(x) = \lg(x) / \lg(10)$
$\ln(x) = \log_{10}(x) / \log_{10}(e)$
That is,
$\log_a(x) = c * \log_b(x)$

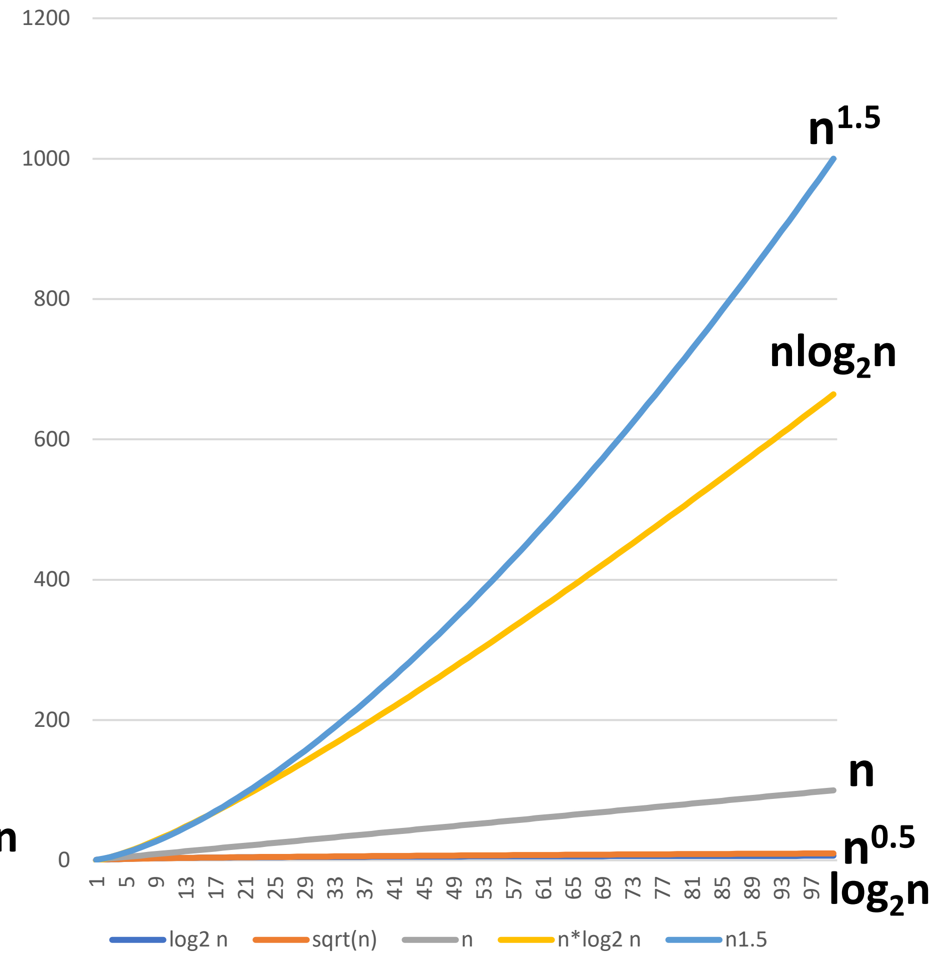
Revision of Complexity Analysis

$f(n) = n^2$, $\log n$, $n\lg n$, $n!$, 2^n, 3^n, $\lg n$, n, $n^{1/2}$, n^5, $\log(n^2)$, 2^{2n}, n^2, 1000, n^n

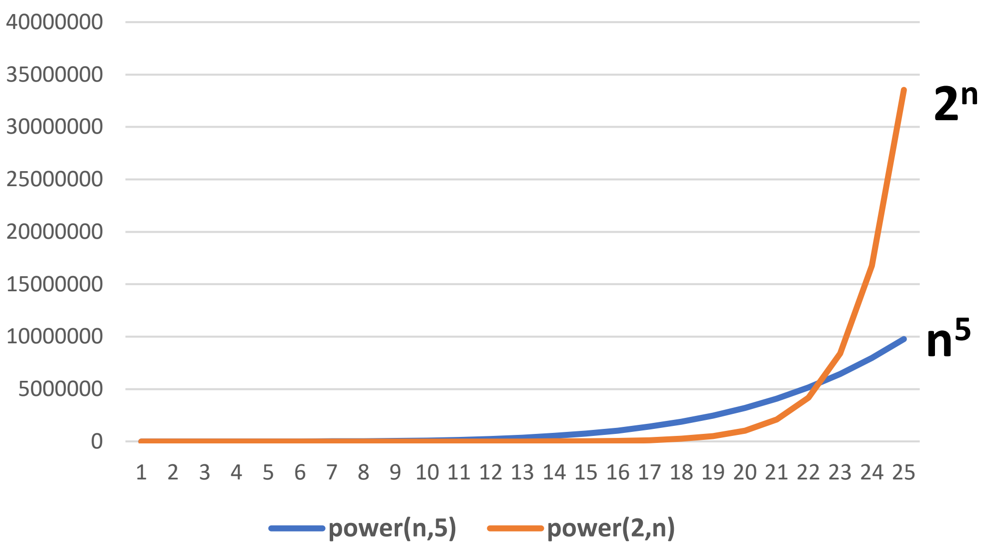
1) $1000 = \theta(1) = O(1)$
2) $\log n$, $\lg n$, $\log(n^2) = \theta(\log n)$
3) $n^{1/2}$
4) n
5) $n\lg n$
6) n^2
7) n^5
8) 2^n
9) 3^n
10) 2^{2n}
11) $n!$
12) n^n

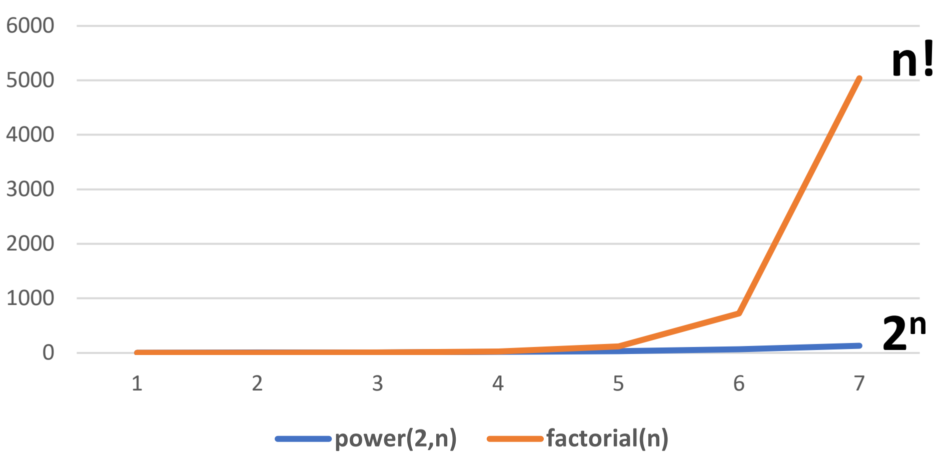
How f(n) increases when n increases





How f(n) increases when n increases





Solving Recurrences (1)

1) The substitution method - guess and check
2) The iteration method - expand (iterate) the recurrence
3) The master method – use the manual

Comments:

- The master method is the easiest for certain types of recurrences and finds the tight bound.

- The substitution method is powerful for almost all types of recurrences and finds the upper bound.

Example of the substitution method

$T(1) = 0$ // worst case of quicksort

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

Guess $T(n) = O(n^2)$ so we try to prove $T(n) <= c*n^2$

Proof:

(1) $T(1) = 0 <= c*1^2$ (so $c=1$)

(2) Assume $T(k) <= k^2$, prove that $T(k+1) <= (k+1)^2$

We have $T(k+1) = T(k) + k$

$$<= k^2 + k <= k^2+2*k+1= (k+1)^2$$

So $T(n) <= n^2$ for all $n >= 1$

Example of the iteration method

Suppose that, instead of using E[middle] as pivot, QuickSort also can use the median of E[first], E[(first + last)/2] and E[last]. How many key comparisons will QuickSort do in the worst case to sort *n* elements? (Remember to count the comparisons done in choosing the pivot.)

Before partition: (P is the median)

P	<P	≥P							

After partition: (3 comparisons to get the median, n-3 comparisons for partition)

<P	P								

$T(1) = 0$

$T(2) = 1$

$T(n) = T(n-2) + n$

Example of the iteration method

$T(1) = 0$
$T(2) = 1$
$T(n) = T(n\text{-}2) + n$

$$T(n) = T(n-2) + n$$
$$= T(n-4) + n - 2 + n$$
$$= T(n-6) + n - 4 + n - 2 + n$$
$$= T(n-8) + n - 6 + n - 4 + n - 2 + n$$

If n = 2k (i.e. n is even)

$T(n) = T(2) + (4 + 6 + 8 + \ldots + n)$ // k-1 terms

$$= 1 + \frac{k-1}{2}(4 + n)$$
$$= O(n^2)$$

$a_i = a_1 + (i\text{-}1)d$

$s_k = \frac{k}{2}(a_1 + a_k)$

Example of the iteration method

If n = 2k+1 (n is odd)

$T(n) = T(1) + (3 + 5 + 7 + \ldots + n)$ // k terms

$= \frac{k}{2}(3 + n)$

$= O(n^2)$

$a_i = a_1 + (i-1)d$

$s_k = \frac{k}{2}(a_1 + a_k)$

Example of the master method

Multiplying two nxn matrices

$W(n) = 7W(n/2) + 15n^2/4$

$n^{\log_b a} = n^{\log_2 7} = n^{\frac{\ln 7}{\ln 2}} = n^{2.8075}$

$$\boxed{\log_b x = \frac{\log_d x}{\log_d b}}$$

$f(n) = 15n^2/4$

$= O(n^{2.8075-0.5})$

$W(n) = \theta(n^{2.8075})$

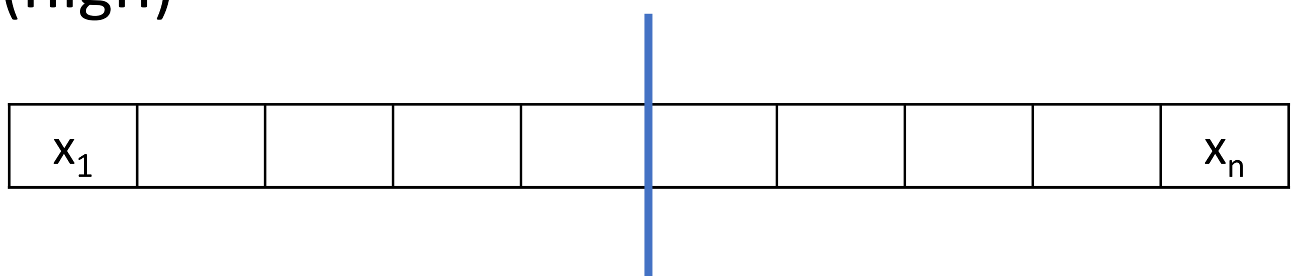
Additional example of the substitution method

Recurrence for the best case of mergesort:

$T(1) = 0$

$T(n) = 2T(n/2) + n/2$

Guess $T(n) = O(n\lg n)$



Proof: consider n is a power of 2.

(1) $T(1) = 0 <= 1*\lg 1$

(2) Assume that $T(2^k) <= k* 2^k$, prove that $T(2^{k+1}) <= (k+1)*2^{k+1}$.

$T(2^{k+1}) = 2T(2^k) + 2^k$

$<= 2* k* 2^k + 2^k$

$<= k* 2^{k+1} + 2^k + 2^k$

$= (k+1)* 2^{k+1}$

Thus $T(n) = O(nlgn)$