Additional Material

- The following slides are for reference and completeness…
- They will **NOT** be examined!

Big-Oh Notation – Part 1

- Given functions $f(n)$ and $g(n)$, we say that $f(n)$ is $O(g(n))$ if there are positive constants c and n_0 such that:

$f(n) \le cg(n)$ for $n \ge n_0$

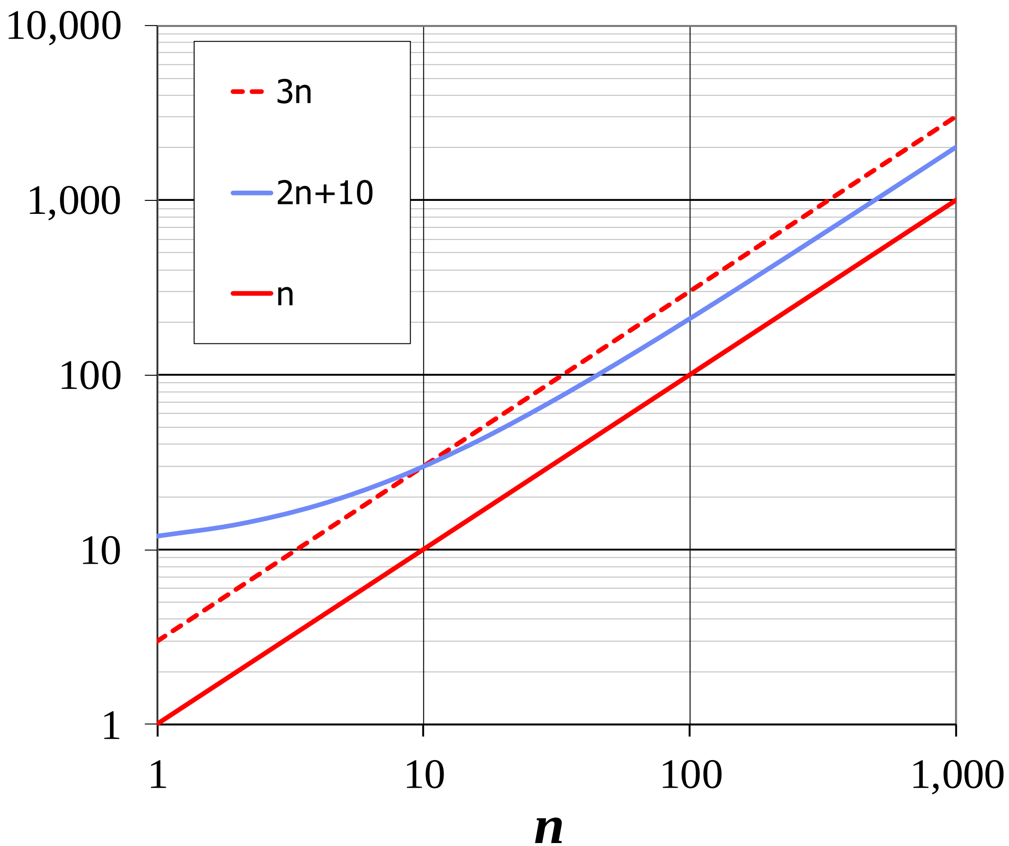
- Is $2n + 10 \equiv O(n)$?
- Can we find c and n_0?

$2n + 10 \le cn$

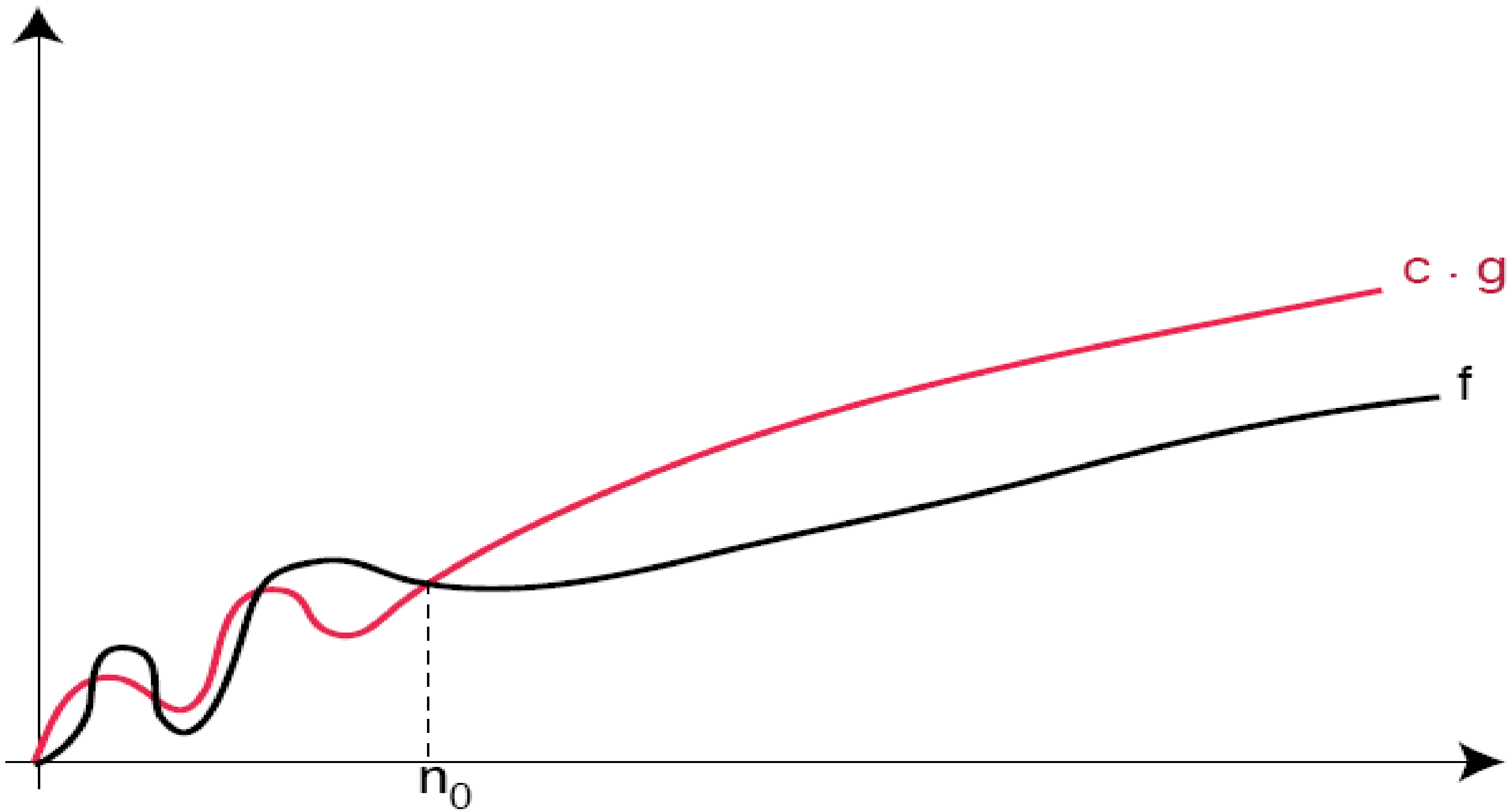
$(c\text{-}2)n \ge 10$

$n \ge 10/(c\text{-}2)$

Pick $c = 3$ and $n_0 = 10$

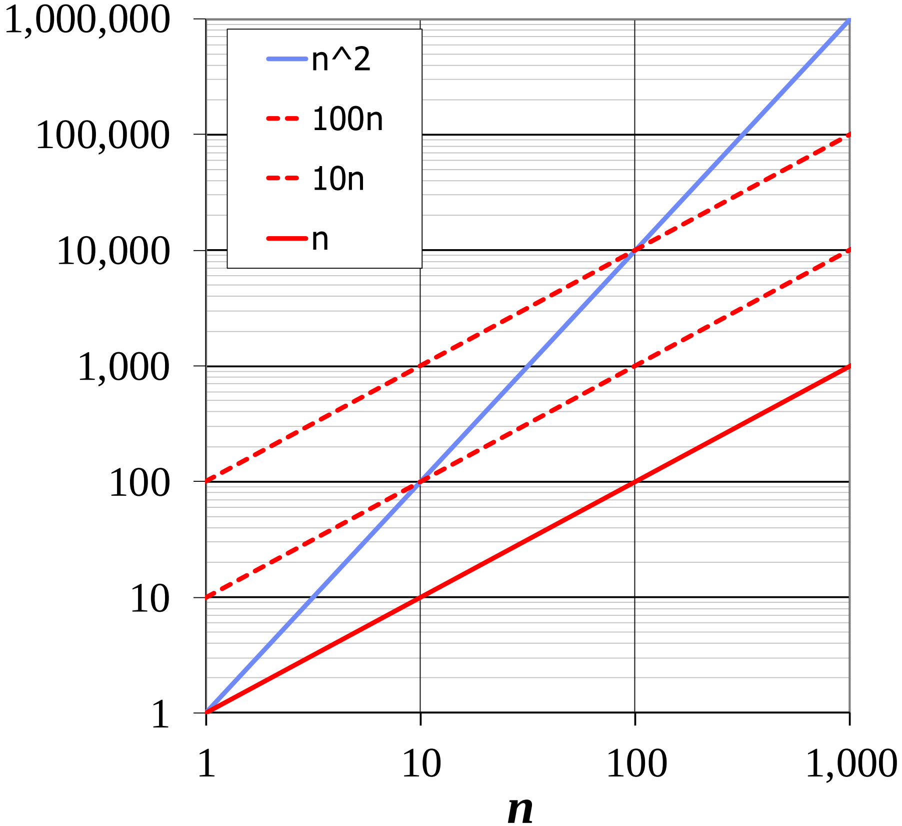


$f(n)$ is $O(g(n))$ *iff* $f(n) \leq cg(n)$ for $n \geq n_0$



Big-Oh Notation – Part 2

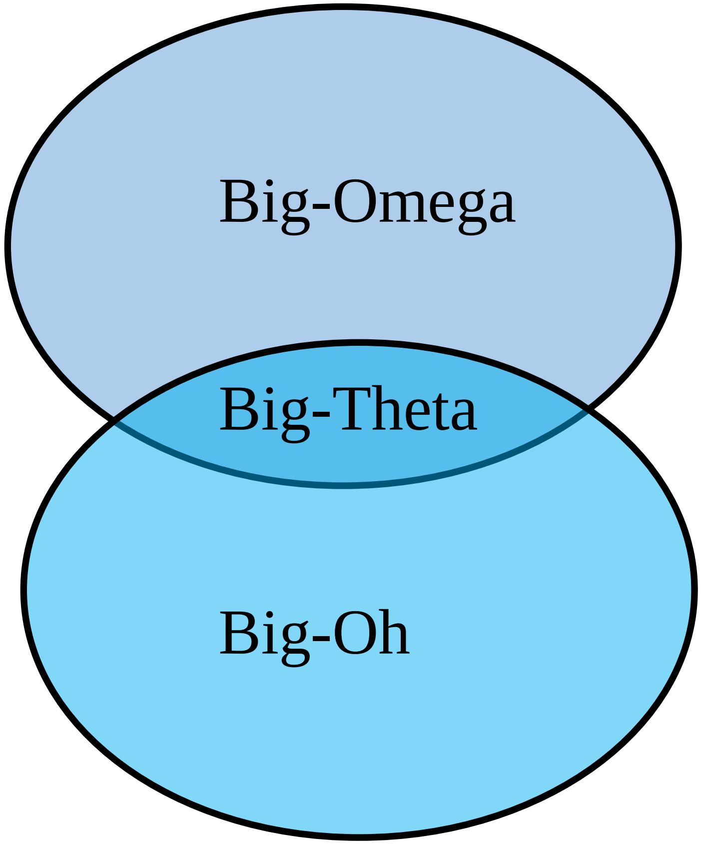
- Example: the function n^2 is not $O(n)$? Why?
  - $n^2 \le cn$
  - $n \le c$
- The above inequality cannot be satisfied since c must be a constant



Big-Oh and Growth Rate

- The Big-Oh notation gives an upper bound on the growth rate of a function
- **The statement "$f(n)$ is $O(g(n))$" means that the growth rate of $f(n)$ is no more than the growth rate of $g(n)$**
- **$f(n)$ grows no faster than $g(n)$**
- We can use the Big-Oh notation to rank functions according to their growth rate

Relatives of Big-Oh

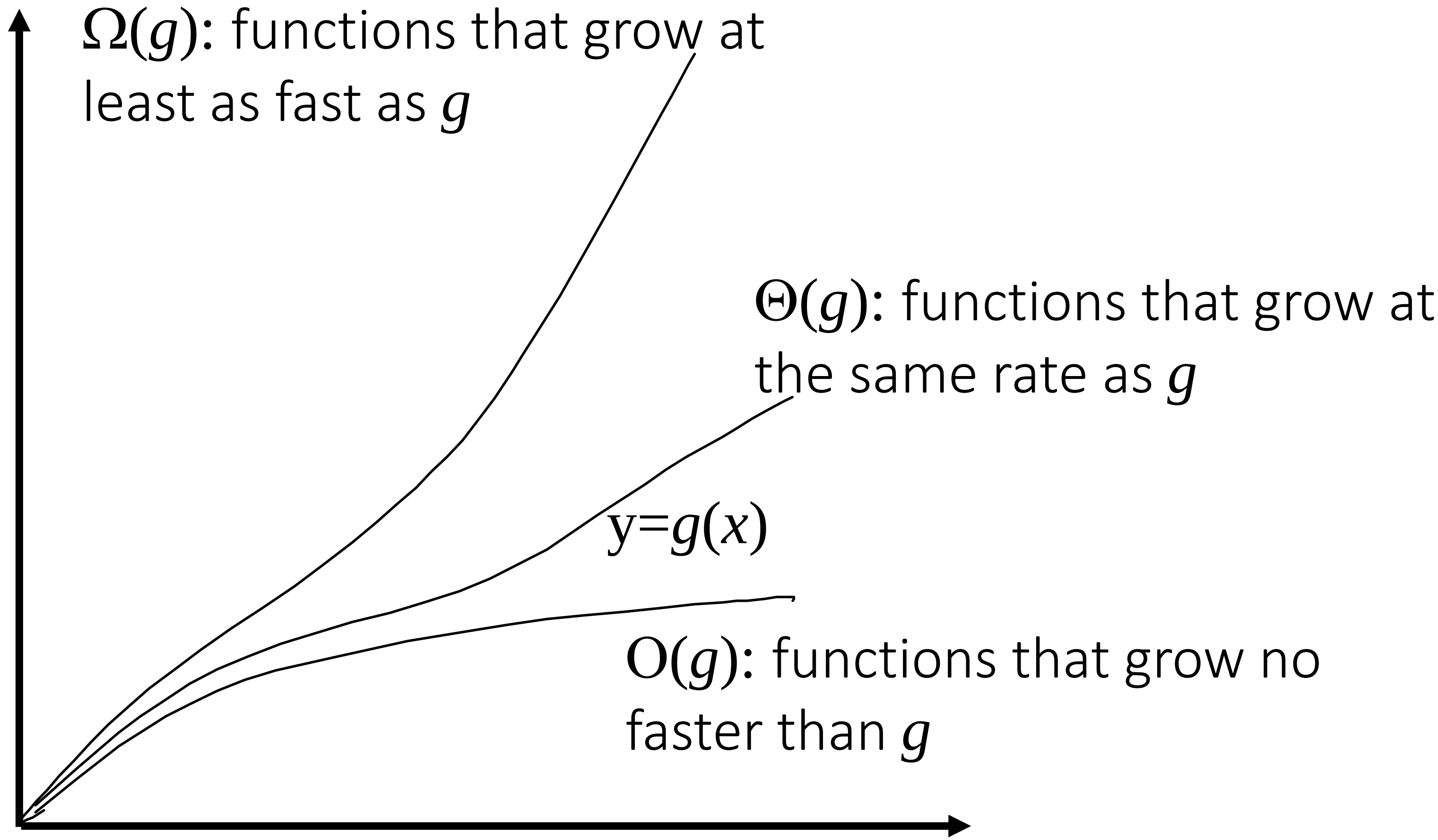


$\Omega(g)$: functions that grow at least as fast as g

$\Theta(g)$: functions that grow at the same rate as g

$O(g)$: functions that grow no faster than g

Asymptotic Notation



Big-Oh is the interesting one

- Because we are interested in efficiency, $\Omega(g)$ will not be of much interest to us because $\Omega(n^2)$ includes all functions that grow faster than n^2, for example, n^3 and 2^n
- For a similar reason, we are not much interested in $\Theta(g)$, the class of functions that grow at the same rate as the function g

- Big-Oh is the class of functions that will be of the greatest interest to us
- Considering two algorithms, we will want to know if the first is in Big-Oh of the second
- If yes, we know that the second algorithm does not do better than the first in solving the problem