INFO 6205
Program Structure and Algorithms

Nik Bear Brown

Big-O

Topics

Big-O

Big- O

- $O(g)$
  - the set of functions that grow no faster than g.
- $g(n)$ describes the worst case behavior of an algorithm that is $O(g)$
- Two additional notations
- $\Omega(g)$
  - the set of functions, f, such that

$$f(n) > c\, g(n)$$

for some constant, c, and $n > N$

Big- O

- *Informally,* Time to solve a problem of size, n,

$$T(n) \quad \text{is} \quad O(\log n)$$

➧ $T(n) = c \log_2 n$

- *Formally:*
  - $O(\,g(n)\,)$ is the set of functions, f, such that

$$f(n) < c\, g(n)$$

  for some constant, $c > 0$, and $n > N$

  - Alternatively, we may write

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} \leq c$$

  and say

Properties of the O notation

- Constant factors may be ignored
  - $\forall\ k > 0,\ kf$ is $O(f)$
- Higher powers grow faster
  - n^r is $O(n^s)$ if $0 \leq r \leq s$
- Fastest growing term dominates a sum
  - If f is $O(g)$, then $f + g$ is $O(g)$
    $eg\ \ an^4 + bn^3$ is $O(n^4)$
- Polynomial's growth rate is determined by leading term
  - If f is a polynomial of degree d,
    then f is $O(n^d)$

Properties of the O notation

- f is $O(g)$ is transitive
  - If f is $O(g)$ and g is $O(h)$ then f is $O(h)$
- Product of upper bounds is upper bound for the product
  - If f is $O(g)$ and h is $O(r)$ then fh is $O(gr)$
- Exponential functions grow faster than powers
  - n^k is $O(b^n)$ $\forall$ $b > 1$ *and* $k \geq 0$
    eg n^{20} *is* $O(1.05^n)$
- Logarithms grow more slowly than powers
  - $\log_b n$ is $O(n^k)$ $\forall$ $b > 1$ *and* $k > 0$
    eg $\log_2 n$ *is* $O(n^{0.5})$