Lecture 18 - More on Big-Oh

<u>Def</u>: $f \in O(g)$ if $\exists c, n_0$ s.t.

$$f(n) \leq c\, g(n) \quad \forall \underline{\underline{n \geq n_0}}$$

limits

Sum, product rules.

<u>Limits rule</u> $f, g \geq 0$

$$\frac{f(n)}{g(n)} \qquad \text{Assume } \lim_{n \to \infty} \frac{f(n)}{g(n)} \text{ exists}$$

a) $\displaystyle\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ then $f \in O(g)$
$g \notin O(f)$

b) $\displaystyle\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$ then $g \in O(f)$
$f \notin O(g)$

c) $\displaystyle\lim_{n \to \infty} \frac{f(n)}{g(n)} = x \neq 0$

"For n big enough $f \approx x \cdot g$" $f \in O(g)$ <u>and</u> $g \in O(f)$

$$g \approx \frac{1}{x} f$$

Proof

Def of limit

Part a: from def $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \Rightarrow$

$\Rightarrow \exists \varepsilon, n_\varepsilon$ s.t. $\forall n \geq n_\varepsilon, \frac{f(n)}{g(n)} \leq \varepsilon$

fix ε

Part a $f \in O(g)$ (const is ε, $n_0 = n_\varepsilon$)

Part c $\lim_{n \to \infty} \frac{f(n)}{g(n)} = x$ $\left| \frac{f(n)}{g(n)} - x \right| < \varepsilon \quad \forall n > n_\varepsilon$

$c = x + \varepsilon \quad n_0 = n_\varepsilon \Rightarrow O()$ def applies

$g \notin O(f)$ $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \Rightarrow \lim_{n \to \infty} \frac{g(n)}{f(n)} = \infty \quad g \notin O(f)$

Example: Prove $\ln n \in O(n)$

$\lim_{n \to \infty} \frac{\ln n}{n} \Rightarrow$ use l'hopital rule

$= \lim_{n \to \infty} \frac{1}{n} = 0$ //

Inclusion (strict) of different $O()$ classes

"Hierarchy" of complexity classes (inclusion)

$O(1)$ - small sequences w assignments, ifs ...

$\cap|$

$O(\log n)$ - Binary Search ($\log_2 n$; any log in any base is in same complexity class) (using limits rule)

structure of size n $\to$ n/k, n/k, $\vdots$, n/k — Recursion on <u>one</u> part

$O(\log n) \subseteq O(n)$ min / max in array, linear search

$O(\log n) \subseteq \underline{O(\sqrt{n})} \subseteq O(n)$

$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}} = 0 \qquad \lim_{n\to\infty} \frac{\sqrt{n}}{n} = 0$ (Exercise)

$$\sqrt{n} = n^{1/2} = \left(2^{\log_2 n}\right)^{1/2} = 2^{\frac{1}{2}\log_2 n}$$

$n \to$ very big <u>integer</u> $\Rightarrow$ represent it in base 2

$\log_2 n$ bits $\Rightarrow$ <u>exponential</u> in no. of bits used to represent the number.

Cryptography/security

$O(n) \subseteq O(n \log n) \to$ MergeSort

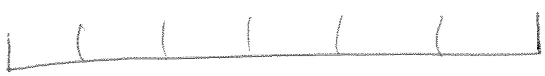
$O(n \log n) \subseteq O(n^2) \to$ BubbleSort, SelectionSort

↳ 2 dimensional arrays (matrices)

$O(n^k) \subseteq O(a^n)$ $\forall k$ constant

$a > 1$ constant

Poly-time   exponential algorithms



n ~~bits~~ positions

a integer; every position can have any value $\{1, 2 \ldots a\}$

Generate and test approach to search for some condition on the array $\Rightarrow a^n$ possibilities

$a < b$ $\quad O(a^n) \subseteq O(b^n)$ (limits rule)

$O(a^n) \subseteq O(n!)$ $\quad n! = 1 \cdot 2 \cdots n$

(Exercise to show $\cancel{O(2^n) \subseteq}$ $2^n \subseteq O(n!)$)

$O(n!)$ - where does it come from?

Permutations

n elements $\Rightarrow$ generate (or count) all possible orderings of ~~these~~ elements

without loss of generality $1 \ldots n$

1 element: $\{1\}$ $\quad P_1 = 1$

2 elements: $\{1, 2\}, \{2, 1\}$ $\quad P_2 = 2 = 1 \cdot 2$

3 elements:

$$\begin{array}{ll} 1, \underline{2\ 3} & 1\ \underline{3\ 2} \\ 2\ \underline{1\ 3} & 2\ \underline{3\ 1} \\ 3\ \underline{1\ 2} & 3\ \underline{2\ 1} \end{array}$$

$P_3 = 6 = 1 \cdot 2 \cdot 3$

$P_3 = 3 \cdot P_2$

In general: n elements: pick each one in turn, shuffle the rest

$$P_n = n \cdot P_{n-1}$$

Guess $P_n = n!$

Proof by induction

Base case: $n = 1 \Rightarrow P_1 = 1!$ true

Ind. step: $P_n = n P_{n-1} \underset{\text{ind hyp}}{=} n \cdot (n-1)! = n!$

Yay!

Randomization

Suppose we want to write an alg works for all shufflings of elements ⇒ shuffle a fixed number of times
(or no. times that is poly in size of struct n)
⇒ run alg ⇒ see answer ⇒ repeat fixed number of times

$O(n!)$ ⇒ approximate algorithm $O(n^k)$

Proof of correctness

Proof that alg is correct with high probability

Finding combinations

n elements, find / count all groups of k elements that are distinct; order does not matter.

$n=1$ $\{1\}$

$n=2$ Groups of 1 element: $\{1\}$ $\{2\}$ $\to 2$

2 elements: $\{1,2\} \to 1$

$\binom{n}{k} \to$ "n choose k": how many groups of k in n elements.

$n=3$, $k=2$ $\binom{3}{2}$

$1 \; \_\_ \in \{2,3\}$ $\binom{2}{1}$

$2 \; \_\_ \in \{1,3\}$ $\binom{2}{1}$

$3 \; \_\_ \in \{1,2\}$ $\binom{2}{1}$

Avoid double-counting: $\{1,2\}, \{2,1\} \to 1$ set.

$3\binom{2}{1}$ but divide further

In general: $\binom{n}{k} = \binom{n-1}{k-1} \cdot \frac{n}{k}$ — all possible choices in 1st place; avoid double counting

Guess $\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$ prove this by induction n