Running Times. Reduction

Friday, January 17, 2025 12:13 PM

Last Time:
- Analysis
- Notations:
  - Big-O
  - Big-Omega
  - Big-Theta

Today:
- Wrap up of Big-O
- First algorithm technique
- "Real" **HW1** will be assigned
  - Due next Friday (24th)
- **No Class Monday!**

$$T(n) = (3n)! = \underbrace{(3n)(3n-1)(3n-2)\cdots(1)}_{3n}$$

upperbound?

$T(n)$ is $\cancel{O(n^n)}$ $\boxed{O(n^{4n})}$

Def $T(n)$ is $O(n^n)$ means that for all $n \geq n_0$,

$$(3n)! \leq c \cdot n^n$$

$$\underbrace{(3n)(3n-1)(3n-2)\cdots(1)} \leq c \cdot n^n$$

$$(3n)^{3n} \leq c \cdot n^n$$

$$3^{3n} \cdot n^{3n} \leq c \cdot n^n$$

$$3^{3n} n^{3n} \leq c \cdot n^{4n}$$

lowerbound for $T(n)$?

$T(n) = (3n)!$

$T(n) = (3n)!$

$T(n)$ is $\Omega(3^n) \rightarrow \Omega(d^n)$ for any d.

This means

$$(3n)! \geq \varepsilon \cdot 3^n \quad \text{for all } n \geq n_0 \text{ and some } \varepsilon$$

Let $n_0 = 2$

$$\underline{(3n)!} \geq \varepsilon \cdot 3^n \quad \text{for all } n \geq 2.$$

Pick some ε to make the expression true!

$3n! \longrightarrow \boxed{\Theta(n!)}$??

Common times

- polynomial time: $T(n)$ is $O(n^d)$ for some d.

- logarithmic time: $T(n)$ is $O(\log n)$.

  $\log^2(n)$ is still asymptotically smaller than n.

$$\log_c n = \frac{\log_{10} n}{\log_{10} c} \rightarrow O(\log_{10} n)$$

- Exponential time: $O(2^n)$

- $O(n!)$

$O(n!)$

logs < poly < exp < factorial

$n \log n$ $\qquad n^{1.5} = n\sqrt{n}$

$1000 n\log n$ v.s. $n^{1.5}$

$1000 n^2$ v.s. n^3

Algorithm Design Techniques

1. Strategies
2. Practice!!

★ Reduction → the most common technique.

1. Given problem X, we solve it by reducing to some other problem Y and solving that. (subroutine).

and solving that. (subroutine).

Peasant Mult

1. Addition
2. Parity
3. doubling
4. halving

→ the details of addition doesn't matter (for correctness)

subroutines

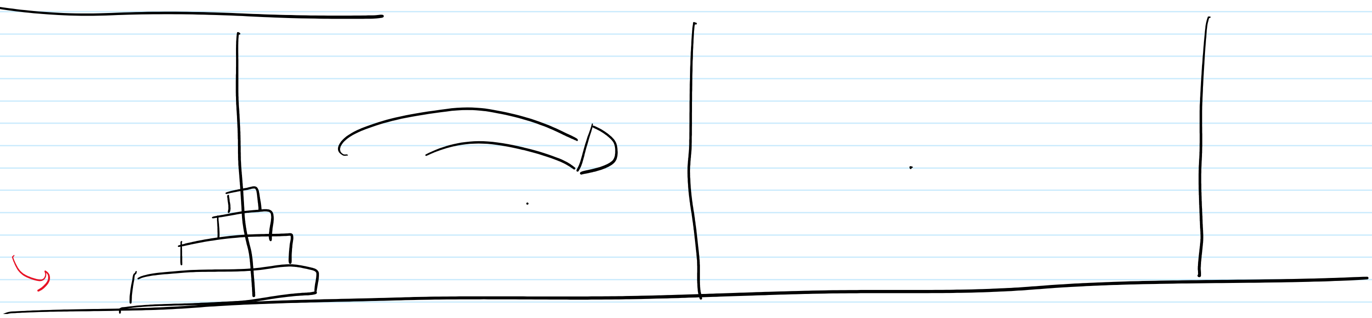
Recursion → "simplify and delegate"

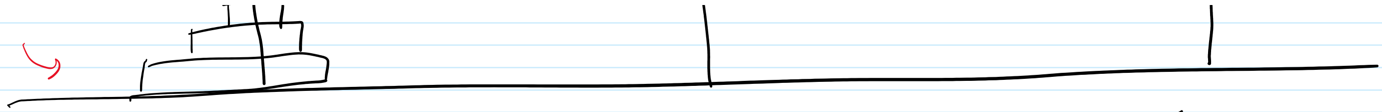
1. If the problem is easy, then solve and find the answer.
2. If the problem is hard, simplify and reduce the problem into one or more instances of the same problem. (but easier)

"induction is recursion (in a different order)"

Standard recursion example

Tower of Hanoi





① move 1 disc at a time

② big discs must be below small discs.

③ Goal: move all discs to the dest.

Easy (if the idea is known).

· Look at the largest disc!

Tower(n, src, dest, tmp)

  if $n \le 0$: do nothing.
  else:
    ① Tower(n-1, src, tmp, dest)
    ② move disc n to dest
    ③ Tower(n-1, tmp, dest, src)

Proof of correctness

Induction

Base case: When $n=0$, the algorithm correctly moves nothing.

Inductive hyp

When there are n-1 discs, my alg. correctly moves them.

My alg. moves n discs correctly by doing the following:

① moves n-1 disc to the tmp.

② move 1 disc.

③ move n-1 discs to dest

All n discs are thus moved correctly to dest