DISCRETE MATHEMATICS

IN COMPUTER SCIENCE

HSIEN-CHIH CHANG

FEBRUARY 25, 2022

Asymptotics and Recurrence

Big-O notation

- $O(g(n))$: a function $f(n)$ such that

$$f(n) \le C \cdot g(n)$$

for some C and for every large enough n

BubbleSort runs in $O(n^2)$ time on an array of size n.

Example

Using $O(g(n))$ in a statement

Example

Using $O(g(n))$ in a statement

$$\log n! = O(n \log n)$$

$$\log n! = n \log n - n + (\log n)/2 + O(1)$$

$$n! = O((n/e)^n)$$

Example

Using $O(g(n))$ in a statement

$$n \log n = O(n^2)$$

Big-Ω and Big-Θ notation

- $\Omega(g(n))$: a function f(n) such that

$$f(n) \geq C \cdot g(n)$$

for some C and for every large enough n

- $\Theta(g(n))$: a function f(n) such that

$$c \cdot g(n) \leq f(n) \leq C \cdot g(n)$$

for some c and C and for every large enough n

Little-o and Little-ω notation

- $o(g(n))$: a function $f(n)$ such that

$$f(n) < C \cdot g(n)$$

for any C and for every large enough n

- $\omega(g(n))$: a function $f(n)$ such that

$$f(n) > C \cdot g(n)$$

for any C and for every large enough n

Example

$n! = \Theta(n^{0.5}(n/e)^n)$

How to solve Recurrence?

Recurrence

Tower of Hanoi

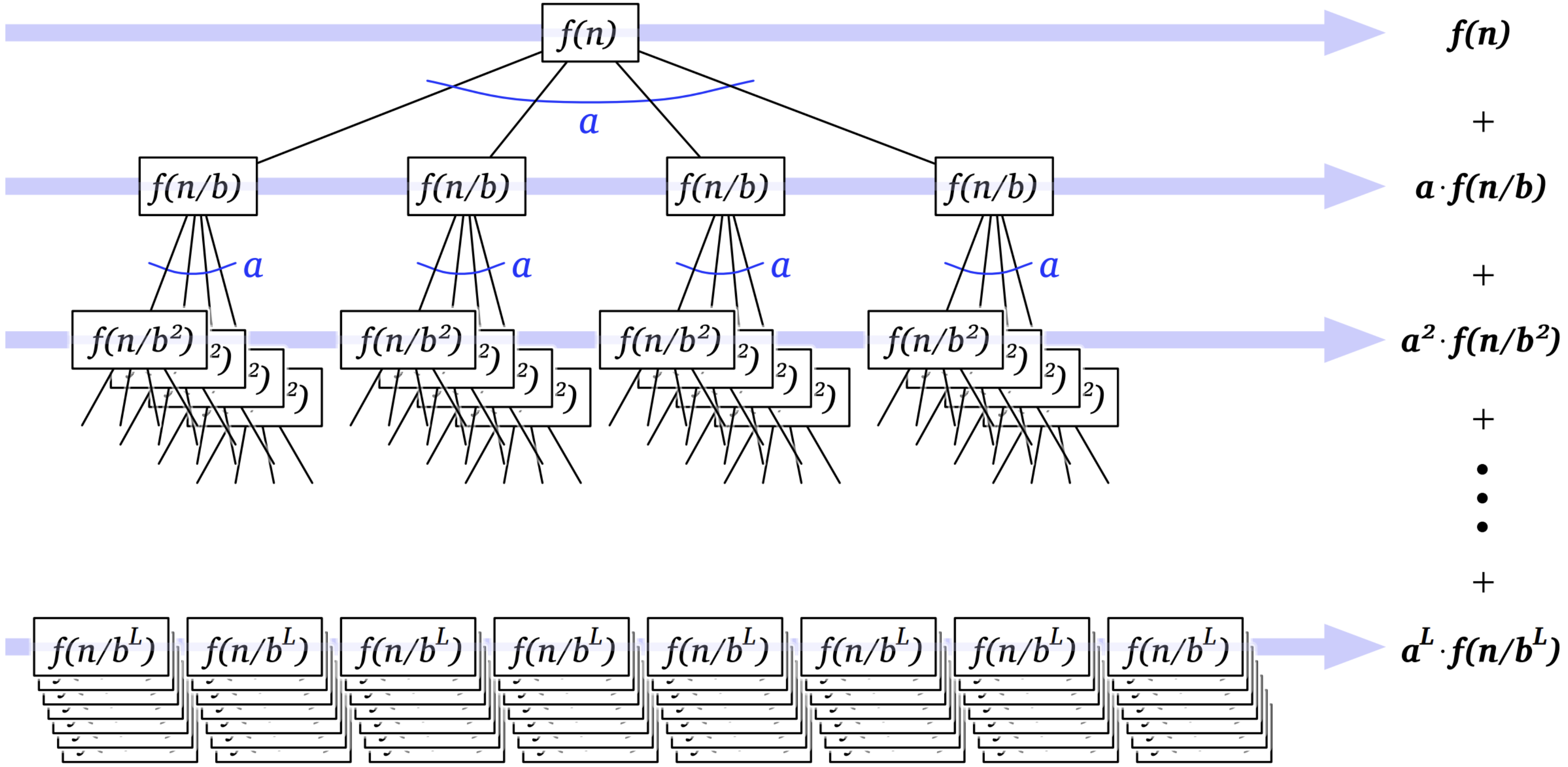
$$T(n) = 2T(n-1) + 1$$

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

Recurrence

MergeSort

Recursion Tree Method



A recursion tree for the recurrence $T(n) = a\,T(n/b) + f(n)$

$$T(n) = T(3n/4) + n$$

Recurrence

MergeSort

Recurrence

Karatsuba's multiplication

$$T(n) = 3T(n/2) + n$$

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

Recurrence

$$T(n) = T(3n/4) + T(n/4) + n$$

Recurrence

Fragile; handle with Care

Next time.

Probability, another fancy name for counting