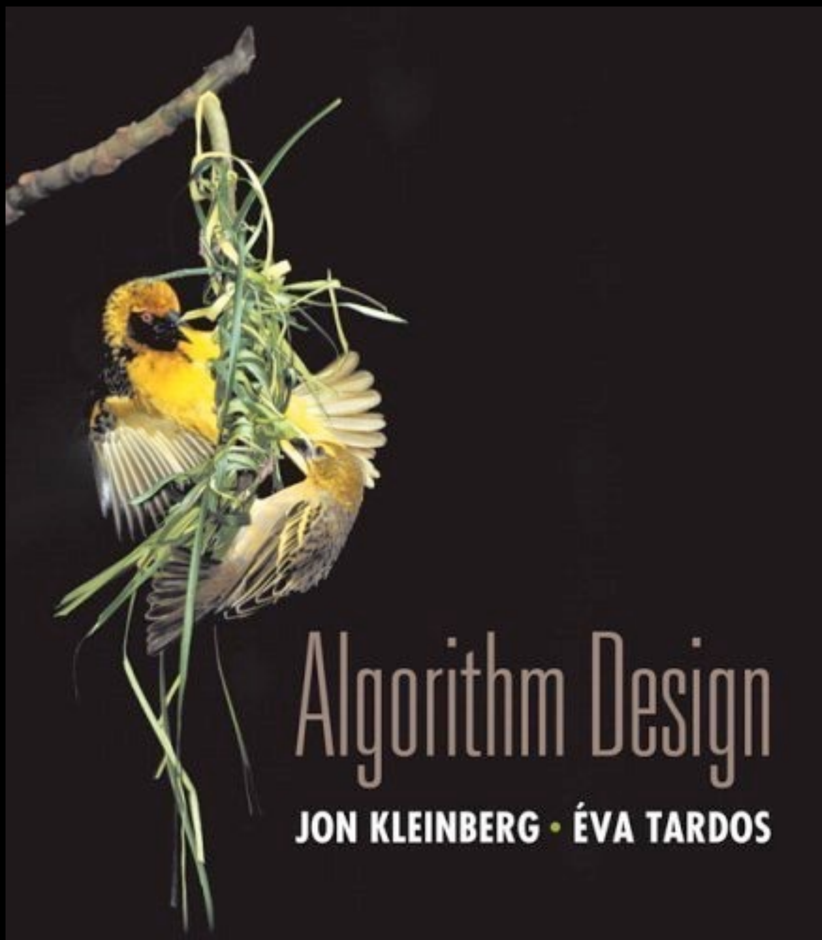


Chapter 13

Randomized Algorithms

Slides by Kevin Wayne.

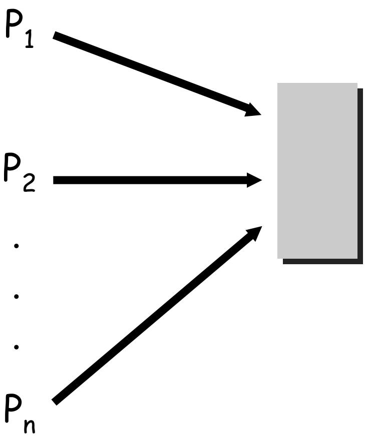
13.1 Contention Resolution

Contention Resolution in a Distributed System

Contention resolution. Given n processes $P_1, \ldots, P_n$, each competing for access to a shared database. If two or more processes access the database simultaneously, all processes are locked out. Devise protocol to ensure all processes get through on a regular basis.

Restriction. Processes can't communicate.

Challenge. Need symmetry-breaking paradigm.



Contention Resolution: Randomized Protocol

Protocol. Each process requests access to the database at time t with probability $p = 1/n$.

Claim. Let S[i, t] = event that process i succeeds in accessing the database at time t. Then $1/(e \cdot n) \le \Pr[S(i, t)] \le 1/(2n)$.

Claim. The probability that process i fails to access the database in en rounds is at most $1/e$. After $e \cdot n(c \ln n)$ rounds, the probability is at most n^{-c}.

Claim. The probability that all processes succeed within $2e \cdot n \ln n$ rounds is at least $1 - 1/n$.

13.3 Linearity of Expectation

Guessing Cards

Game. Shuffle a deck of n cards; turn them over one at a time; try to guess each card.

Memoryless guessing. No psychic abilities; can't even remember what's been turned over already. Guess a card from full deck uniformly at random.

Claim. The expected number of correct guesses is 1.

Guessing with memory. Guess a card uniformly at random from cards not yet seen.

Claim. The expected number of correct guesses is $\Theta(\log n)$.

Coupon Collector

Coupon collector. Each box of cereal contains a coupon. There are n different types of coupons. Assuming all boxes are equally likely to contain each coupon, how many boxes before you have ≥ 1 coupon of each type?

Claim. The expected number of steps is $\Theta(n \log n)$.

13.5 Randomized Divide-and-Conquer

Quicksort

Sorting. Given a set of n distinct elements S, rearrange them in ascending order.

```
RandomizedQuicksort(S) {
   if |S| = 0 return

   choose a splitter aᵢ ∈ S uniformly at random
   foreach (a ∈ S) {
      if      (a < aᵢ) put a in S⁻
      else if (a > aᵢ) put a in S⁺
   }
   RandomizedQuicksort(S⁻)
   output aᵢ
   RandomizedQuicksort(S⁺)
}
```

Remark. Can implement in-place.

↑ $O(\log n)$ extra space

Quicksort

Running time.

- [Best case.] Select the median element as the splitter: quicksort makes $\Theta(n \log n)$ comparisons.
- [Worst case.] Select the smallest element as the splitter: quicksort makes $\Theta(n^2)$ comparisons.

Randomize. Protect against worst case by choosing splitter at random.

Intuition. If we always select an element that is bigger than 25% of the elements and smaller than 25% of the elements, then quicksort makes $\Theta(n \log n)$ comparisons.

Notation. Label elements so that $x_1 < x_2 < \ldots < x_n$.

Quicksort: Expected Number of Comparisons

Theorem. Expected # of comparisons is $O(n \log n)$.

Theorem. [Knuth 1973] Stddev of number of comparisons is ~ $0.65n$.

Ex. If n = 1 million, the probability that randomized quicksort takes less than $4n \ln n$ comparisons is at least 99.94%.

Chebyshev's inequality. $\Pr[|X - \mu| \geq k\delta] \leq 1 / k^2$.

Quicksort: Expected Number of Comparisons

The expected number of comparisons in a randomized Quicksort of n elements is (γ is Euler's constant near 0.577) :

$$q_n = 2n \ln n - (4 - 2\gamma)n + 2 \ln n + O(1).$$

In 1996, McDiarmid and Hayward have formulated an exact expression for the probability that the number of comparisons $\boldsymbol{Q_n}$ be far from its average $\boldsymbol{q_n}$

$$\Pr\left[\left|\frac{Q_n}{q_n} - 1\right| > \varepsilon\right] = n^{-(2+o(1))\varepsilon \ln^{(2)} n}$$

Let c be a positive constant. McDiarmid and Hayward's formula imply that there exists another positive constant a smaller than 1 such that

$$\Pr[\ \boldsymbol{Q_n} \in \Theta(n^{1+c})\ ] < a^{n^c}.$$