# *Algorithm Design and Analysis*

CSE 565

## LECTURE 42

**Randomized Algorithms**

- Contention Resolution
- Global Min-Cut
- Linearity of Expectation (not covered in class)

**Sofya Raskhodnikova**

12/10/2007

*S. Raskhodnikova; based on slides by K. Wayne.*

---

## Randomization

Algorithmic design patterns.
- Greedy.
- Divide-and-conquer.
- Dynamic programming.
- Network flow.
- Randomization.

in practice, access to a pseudo-random number generator

Randomization. Allow fair coin flip in unit time.

Why randomize? Can lead to simplest, fastest, or only known algorithm for a particular problem.

Ex. Symmetry breaking protocols, graph algorithms, quicksort, hashing, load balancing, Monte Carlo integration, cryptography.

---

## 13.1 Contention Resolution

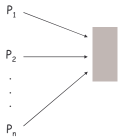
---

## 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 $1/(e \cdot n) \le \Pr[S(i, t)] \le 1/(2n)$.

Pf. By independence, $\Pr[S(i, t)] = \overset{\text{process i requests access}}{p}\ \overset{\text{none of remaining n-1 processes request access}}{(1-p)^{n-1}}$.

- Setting $\underset{\text{value that maximizes Pr[S(i, t)]}}{p = 1/n}$, we have $\Pr[S(i, t)] = 1/n\ \underbrace{(1 - 1/n)^{n-1}}_{\text{between 1/e and 1/2}}$. ▪

Useful facts from calculus. As n increases from 2, the function:
- $(1 - 1/n)^n$ converges monotonically from 1/4 up to 1/e
- $(1 - 1/n)^{n-1}$ converges monotonically from 1/2 down to 1/e.

---

## Contention Resolution: Randomized Protocol

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 $n^{-c}$.

Pf. Let F[i, t] = event that process i fails to access database in rounds 1 through t. By independence and previous claim, we have $\Pr[F(i, t)] \le (1 - 1/(en))^t$.

- Choose $t = \lceil e \cdot n \rceil$: $\Pr[F(i,t)] \le \left(1 - \frac{1}{en}\right)^{\lceil en \rceil} \le \left(1 - \frac{1}{en}\right)^{en} \le \frac{1}{e}$
- Choose $t = \lceil e \cdot n \rceil \lceil c \ln n \rceil$: $\Pr[F(i, t)] \le \left(\frac{1}{e}\right)^{c \ln n} = n^{-c}$

## Contention Resolution: Randomized Protocol

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

Pf. Let F[t] = event that at least one of the n processes fails to access database in any of the rounds 1 through t.

$$\Pr\big[F[t]\big] \;=\; \Pr\Big[\bigcup_{i=1}^{n} F[i,t]\Big] \;\underset{\text{union bound}}{\le}\; \sum_{i=1}^{n} \Pr[F[i,t]] \;\underset{\text{previous slide}}{\le}\; n\left(1-\tfrac{1}{en}\right)^{t}$$

- Choosing $t = 2\lceil en\rceil\lceil c \ln n\rceil$ yields $\Pr[F[t]] \le n \cdot n^{-2} = 1/n$. ▪

Union bound. Given events $E_1, \ldots, E_n$, $\Pr\Big[\bigcup_{i=1}^{n} E_i\Big] \le \sum_{i=1}^{n} \Pr[E_i]$

## 13.2 Global Minimum Cut

## Global Minimum Cut

Global min cut. Given a connected, undirected graph G = (V, E) find a cut (A, B) of minimum cardinality.

Applications. Partitioning items in a database, identify clusters of related documents, network reliability, network design, circuit design, TSP solvers.

Network flow solution.

- Replace every edge (u, v) with two antiparallel edges (u, v) and (v, u).
- Pick some vertex s and compute min s-v cut separating s from each other vertex $v \in V$.

False intuition. Global min-cut is harder than min s-t cut.

## Contraction Algorithm

Contraction algorithm. [Karger 1995]

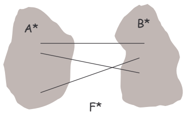
- Pick an edge e = (u, v) uniformly at random.
- Contract edge e.
  - replace u and v by single new super-node w
  - preserve edges, updating endpoints of u and v to w
  - keep parallel edges, but delete self-loops
- Repeat until graph has just two nodes $v_1$ and $v_2$.
- Return the cut (all nodes that were contracted to form $v_1$).

## Contraction Algorithm

Claim. The contraction algorithm returns a min cut with prob $\ge 2/n^2$.

Pf. Consider a global min-cut (A*, B*) of G. Let F* be edges with one endpoint in A* and the other in B*. Let k = |F*| = size of min cut.

- In first step, algorithm contracts an edge in F* probability k / |E|.
- Every node has degree $\ge k$ since otherwise (A*, B*) would not be min-cut. $\Rightarrow$ $|E| \ge \frac{1}{2}kn$.
- Thus, algorithm contracts an edge in F* with probability $\le 2/n$.



## Contraction Algorithm

Claim. The contraction algorithm returns a min cut with prob $\ge 2/n^2$.

Pf. Consider a global min-cut (A*, B*) of G. Let F* be edges with one endpoint in A* and the other in B*. Let k = |F*| = size of min cut.

- Let G' be graph after j iterations. There are n' = n-j supernodes.
- Suppose no edge in F* has been contracted. The min-cut in G' is still k.
- Since value of min-cut is k, $|E'| \ge \frac{1}{2}kn'$.
- Thus, algorithm contracts an edge in F* with probability $\le 2/n'$.

- Let $E_j$ = event that an edge in F* is not contracted in iteration j.

$$\begin{aligned}
\Pr[E_1 \cap E_2 \cdots \cap E_{n-2}] &= \Pr[E_1] \times \Pr[E_2 \mid E_1] \times \cdots \times \Pr[E_{n-2} \mid E_1 \cap E_2 \cdots \cap E_{n-3}] \\
&\ge \left(1-\tfrac{2}{n}\right)\left(1-\tfrac{2}{n-1}\right)\cdots\left(1-\tfrac{2}{4}\right)\left(1-\tfrac{2}{3}\right) \\
&= \left(\tfrac{n-2}{n}\right)\left(\tfrac{n-3}{n-1}\right)\cdots\left(\tfrac{2}{4}\right)\left(\tfrac{1}{3}\right) \\
&= \tfrac{2}{n(n-1)} \\
&\ge \tfrac{2}{n^2}
\end{aligned}$$

## Contraction Algorithm

Amplification. To amplify the probability of success, run the contraction algorithm many times and output the best cut found.

Claim. If we repeat the contraction algorithm $n^2 \ln n$ times with independent random choices, the probability of failing to find the global min-cut is at most $1/n^2$.

Pf. By independence, the probability of failure is at most

$$\left(1-\frac{2}{n^2}\right)^{n^2 \ln n} = \left[\left(1-\frac{2}{n^2}\right)^{\frac{1}{2}n^2}\right]^{2\ln n} \underset{\underset{(1 - 1/x)^x \leq 1/e}{\uparrow}}{\leq} \left(e^{-1}\right)^{2\ln n} = \frac{1}{n^2}$$

## Global Min Cut: Context

Best known deterministic. [Nagamochi-Ibaraki 1992] $O(mn + n^2 \log n)$.

Remark. Our algorithm is slow since we perform $\Theta(n^2 \log n)$ iterations and each takes $\Omega(m)$ time.

Improvement. [Karger-Stein 1996] $O(n^2 \log^3 n)$.

- Early iterations are less risky than later ones: probability of contracting an edge in min cut hits 50% when $n / \sqrt{2}$ nodes remain.
- Run contraction algorithm until $n / \sqrt{2}$ nodes remain.
- Run contraction algorithm twice on resulting graph, and return best of two cuts.

Extensions. Naturally generalizes to handle positive weights.

Best known. [Karger 2000] $O(m \log^3 n)$.

↖ faster than best known max flow algorithm
or deterministic global min cut algorithm

## 13.3 Linearity of Expectation

## Expectation

Expectation. Given a discrete random variables X, its expectation E[X] is defined by:

$$E[X] = \sum_{j=0}^{\infty} j \Pr[X = j]$$

Waiting for a first success. Coin is heads with probability p and tails with probability 1-p. How many independent flips X until first heads?

$$E[X] = \sum_{j=0}^{\infty} j \cdot \Pr[X = j] = \sum_{j=0}^{\infty} j \underset{\underset{\text{j-1 tails}}{\uparrow}}{(1-p)^{j-1}} \underset{\underset{\text{1 head}}{\uparrow}}{p} = \frac{p}{1-p} \sum_{j=0}^{\infty} j (1-p)^j = \frac{p}{1-p} \cdot \frac{1-p}{p^2} = \frac{1}{p}$$

## Expectation: Two Properties

Useful property. If X is a 0/1 random variable, E[X] = Pr[X = 1].

Pf. $E[X] = \sum_{j=0}^{\infty} j \cdot \Pr[X = j] = \sum_{j=0}^{1} j \cdot \Pr[X = j] = \Pr[X = 1]$

not necessarily independent

Linearity of expectation. Given two random variables X and Y defined over the same probability space, E[X + Y] = E[X] + E[Y].

Decouples a complex calculation into simpler pieces.

## 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.

Pf. (surprisingly effortless using linearity of expectation)

- Let $X_i$ = 1 if $i^{th}$ prediction is correct and 0 otherwise.
- Let X = number of correct guesses = $X_1 + \ldots + X_n$.
- $E[X_i] = \Pr[X_i = 1] = 1/n$.
- $E[X] \underset{\underset{\text{linearity of expectation}}{\uparrow}}{=} E[X_1] + \ldots + E[X_n] = 1/n + \ldots + 1/n = 1.$ ▪

## Guessing Cards

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

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

**Claim.** The expected number of correct guesses is Θ(log n).

**Pf.**

- Let $X_i = 1$ if $i^{th}$ prediction is correct and 0 otherwise.
- Let X = number of correct guesses = $X_1 + \ldots + X_n$.
- $E[X_i] = \Pr[X_i = 1] = 1 / (n - i - 1)$.
- $E[X] \underset{\substack{\uparrow \\ \text{linearity of expectation}}}{=} E[X_1] + \ldots + E[X_n] = 1/n + \ldots + 1/2 + 1/1 = \underset{\substack{\uparrow \\ \ln(n+1) < H(n) < 1 + \ln n}}{H(n)}$. ▪

## 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 Θ(n log n).

**Pf.**

- Phase j = time between j and j+1 distinct coupons.
- Let $X_j$ = number of steps you spend in phase j.
- Let X = number of steps in total = $X_0 + X_1 + \ldots + X_{n-1}$.

$$E[X] = \sum_{j=0}^{n-1} E[X_j] \underset{\substack{\uparrow \\ \text{prob of success} = (n-j)/n \\ \Rightarrow \text{ expected waiting time} = n/(n-j)}}{=} \sum_{j=0}^{n-1} \frac{n}{n-j} = n \sum_{i=1}^{n} \frac{1}{i} = n H(n)$$