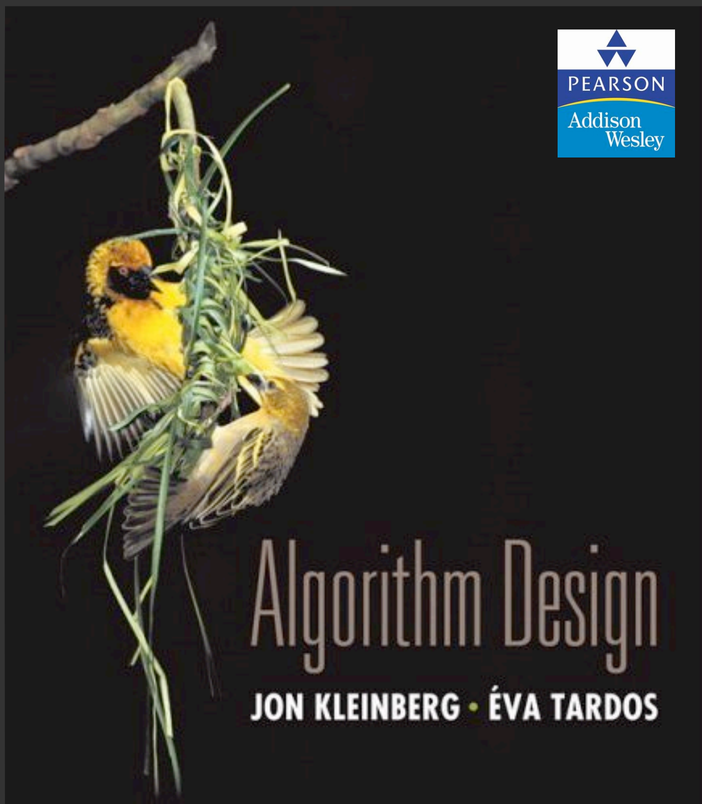


Lecture slides by Kevin Wayne

http://www.cs.princeton.edu/~wayne/kleinberg-tardos

13. Randomized Algorithms

- *content resolution*
- *global min cut*
- *linearity of expectation*
- *max 3-satisfiability*
- *universal hashing*
- *Chernoff bounds*
- *load balancing*

Last updated on Sep 8, 2013 6:54 AM

Randomization

Algorithmic design patterns.

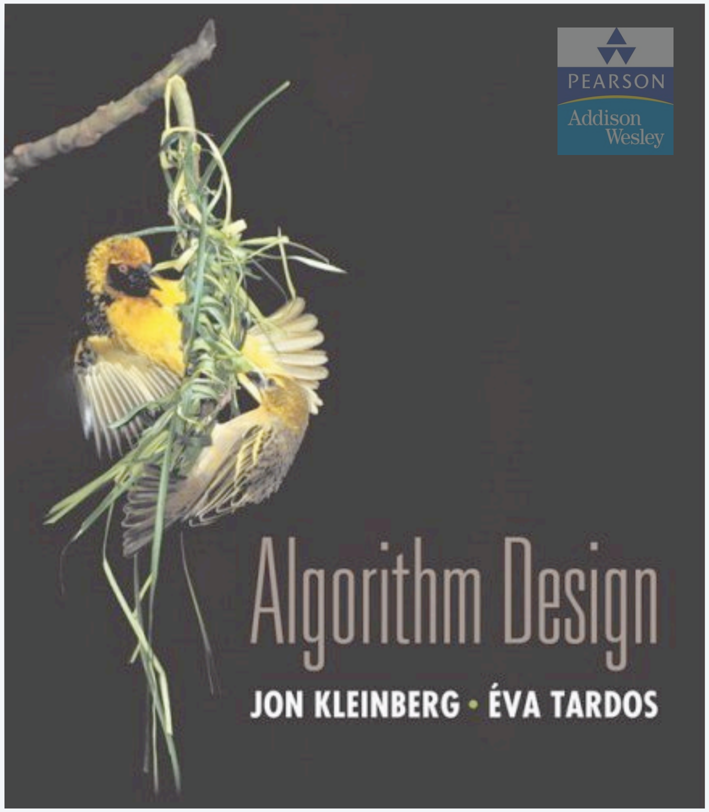
- Greedy.
- Divide-and-conquer.
- Dynamic programming.
- Network flow.
- Randomization.

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.



SECTION 13.1

13. RANDOMIZED ALGORITHMS

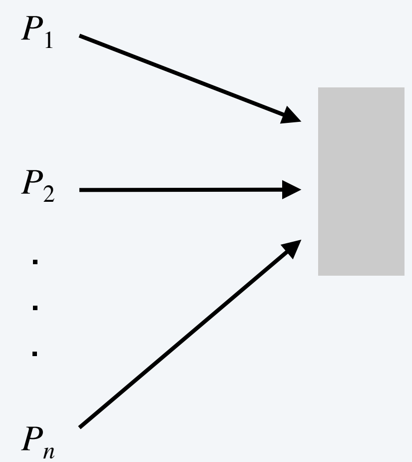
- *content resolution*
- *global min cut*
- *linearity of expectation*
- *max 3-satisfiability*
- *universal hashing*
- *Chernoff bounds*
- *load balancing*

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

Pf. By independence, $\Pr[S(i,t)] = p\,(1-p)^{n-1}$.

process i requests access

none of remaining n-1 processes request access

- Setting $p = 1/n$, we have $\Pr[S(i,t)] \;=\; 1/n\,(1-1/n)^{n-1}$. ▪

value that maximizes Pr[S(i, t)]

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 $\leq n^{-c}$.

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
$\Pr\,[F[i, t]] \;\leq\; (1 - 1/(en))^{\,t}$.

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

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

Contention Resolution: randomized protocol

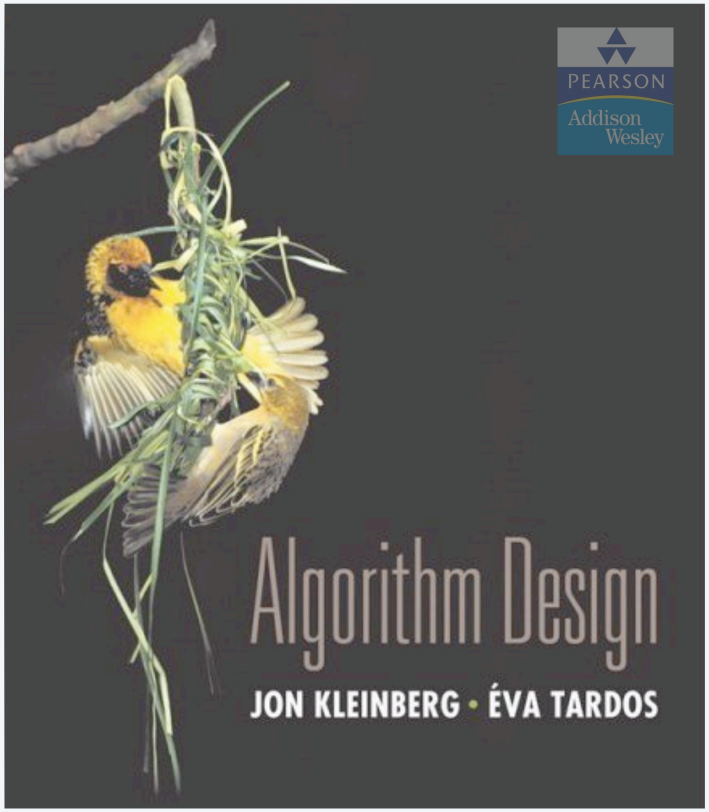
Claim. The probability that all processes succeed within $2e \cdot n \ln n$ rounds is $\geq 1 - 1/n$.

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.

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

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

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



SECTION 13.2

13. RANDOMIZED ALGORITHMS

- *content resolution*
- *global min cut*
- *linearity of expectation*
- *max 3-satisfiability*
- *universal hashing*
- *Chernoff bounds*
- *load balancing*

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.

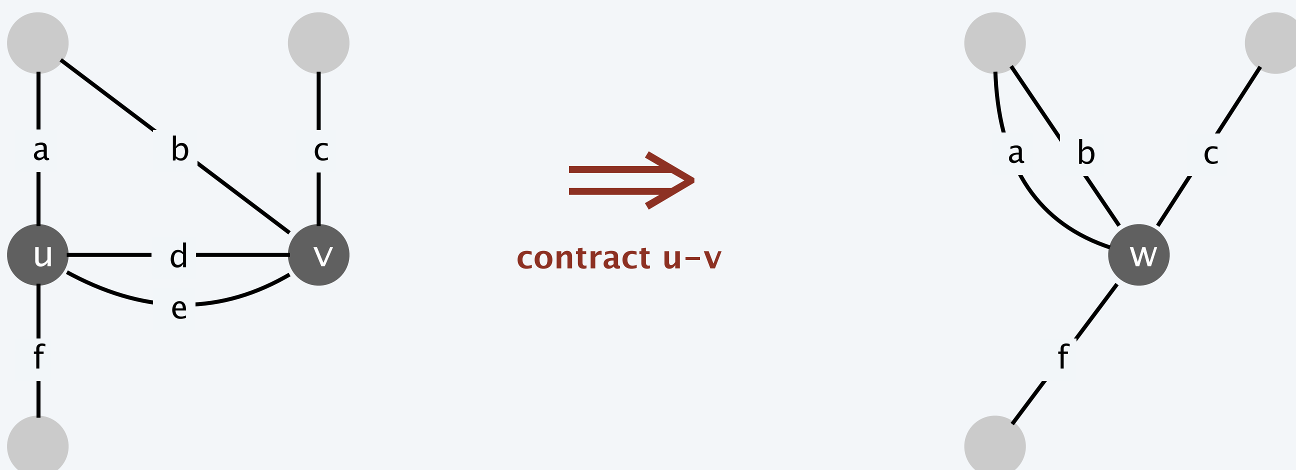
- Replace every edge (u, v) with two antiparallel edges (u, v) and (v, u).
- Pick some vertex s and compute min s- v cut separating s from each other vertex $v \in V$.

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

Contraction algorithm

Contraction algorithm. [Karger 1995]

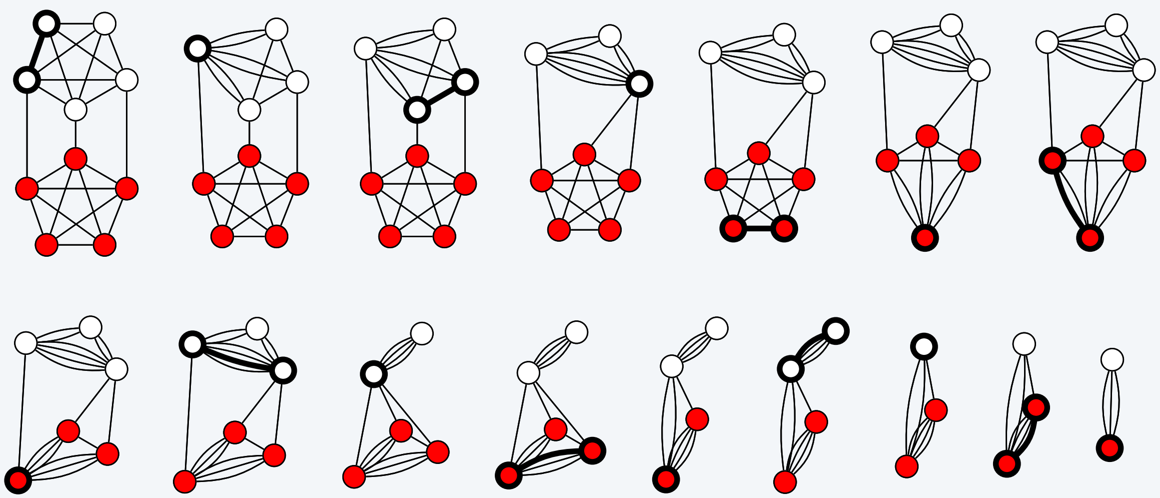
- Pick an edge $e = (u, v)$ uniformly at random.
- Contract edge e.
  - replace u and v by single new super-node w
  - preserve edges, updating endpoints of u and v to w
  - keep parallel edges, but delete self-loops
- Repeat until graph has just two nodes v_1 and v_1.
- Return the cut (all nodes that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



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_1.
- Return the cut (all nodes that were contracted to form v_1).



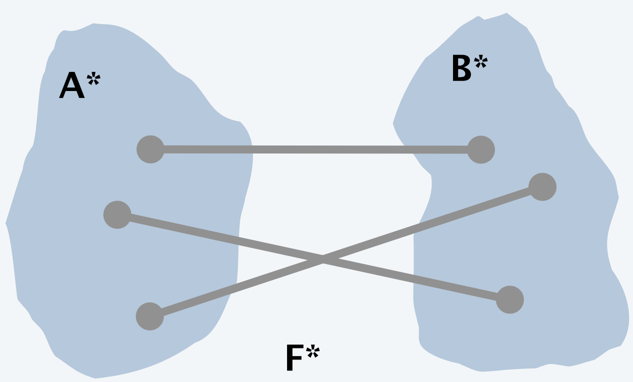
Reference: Thore Husfeldt

Contraction algorithm

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

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 $\geq k$ since otherwise (A^*, B^*) would not be a min-cut $\Rightarrow |E| \geq \frac{1}{2} k n$.
- Thus, algorithm contracts an edge in F^* with probability $\leq 2 / n$.



Contraction algorithm

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

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, $|E'| \geq \frac{1}{2} k n'$.
- Thus, algorithm contracts an edge in F^* with probability $\leq 2 / n'$.
- Let E_j = event that an edge in F^* is not contracted in iteration j.

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

Contraction algorithm

Amplification. To amplify the probability of success, run the contraction algorithm many times.

with independent random choices,

Claim. If we repeat the contraction algorithm $n^2 \ln n$ times, then the probability of failing to find the global min-cut is $\leq 1 / n^2$.

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

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

$(1 - 1/x)^x \leq 1/e$

Contraction algorithm: example execution

trial 1

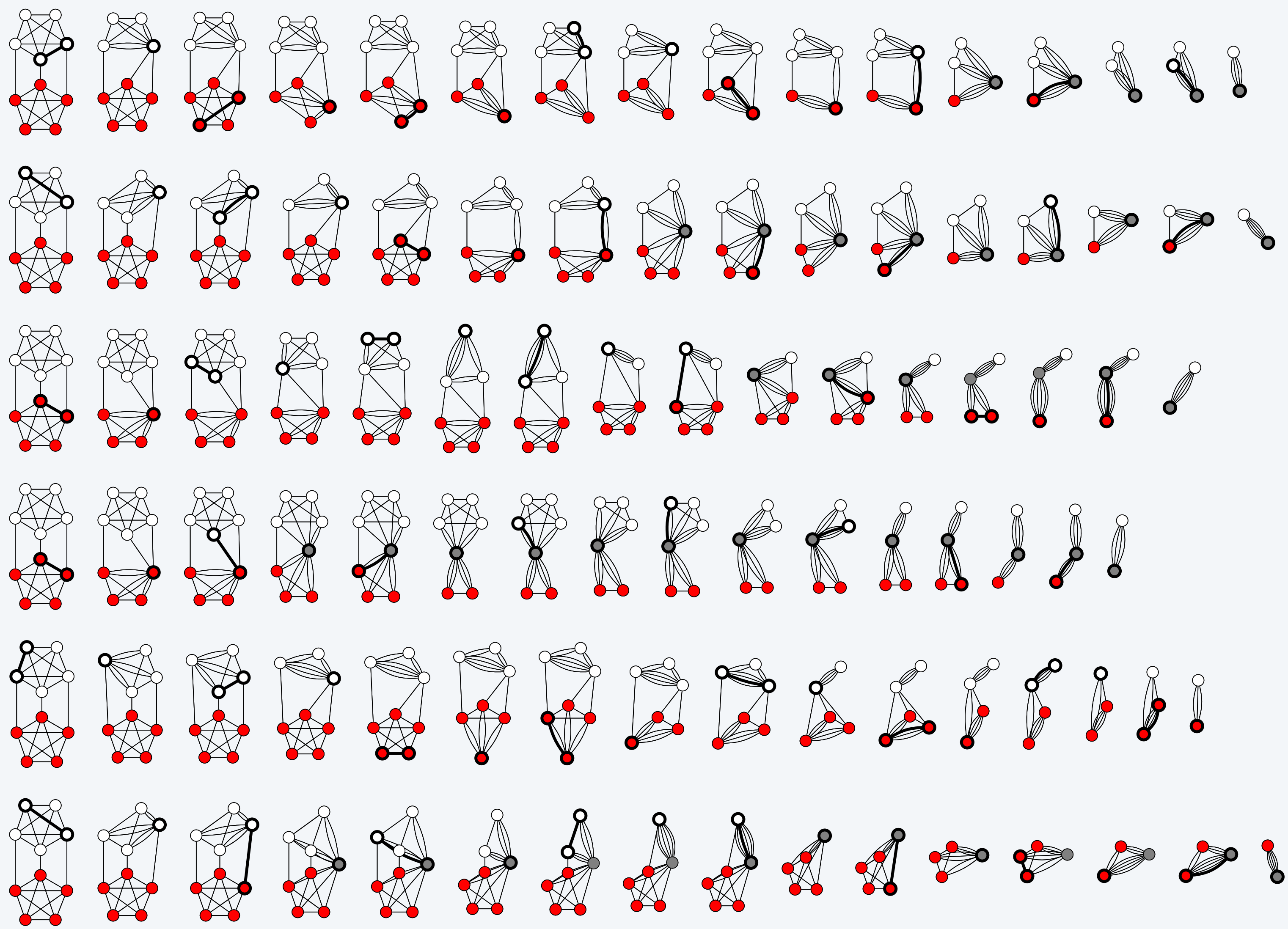
trial 2

trial 3

trial 4

trial 5
(finds min cut)

trial 6



...

Reference: Thore Husfeldt

Global min cut: context

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

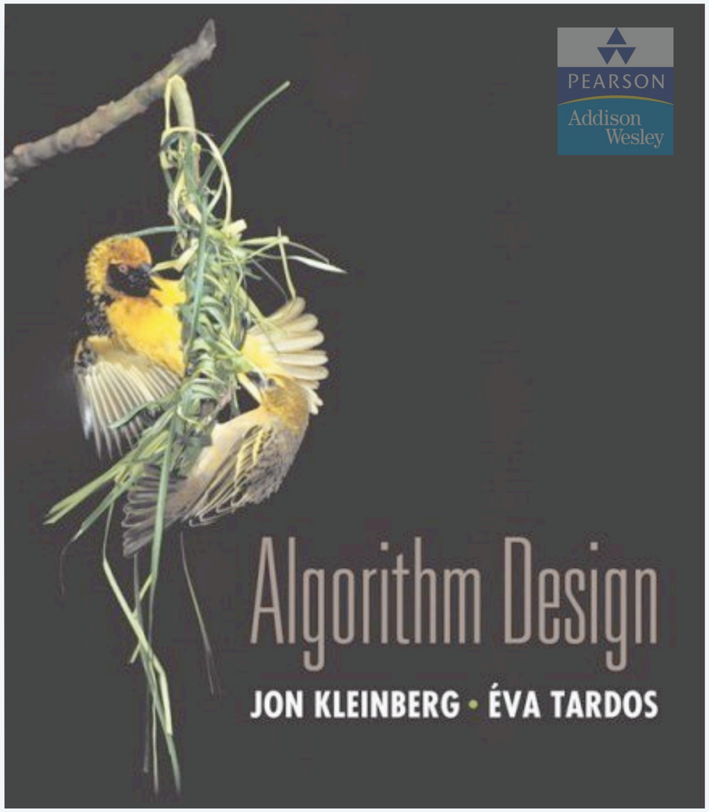
Improvement. [Karger-Stein 1996] $O(n^2 \log^3 n)$.

- 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



SECTION 13.3

13. RANDOMIZED ALGORITHMS

- *content resolution*
- *global min cut*
- *linearity of expectation*
- *max 3-satisfiability*
- *universal hashing*
- *Chernoff bounds*
- *load balancing*

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?

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

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, $E[X + Y] = E[X] + E[Y]$.

Benefit. 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 $= X_1 + \ldots + X_n$.
- $E[X_i] = \Pr[X_i = 1] = 1 / n$.
- $E[X] \underset{\uparrow}{=} E[X_1] + \ldots + E[X_n] = 1/n + \ldots + 1/n = 1$. ▪

linearity of expectation

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 $\Theta(\log n)$.

Pf.

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

linearity of expectation

$\ln(n+1) < H(n) < 1 + \ln n$

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 $\Theta(n \log n)$.

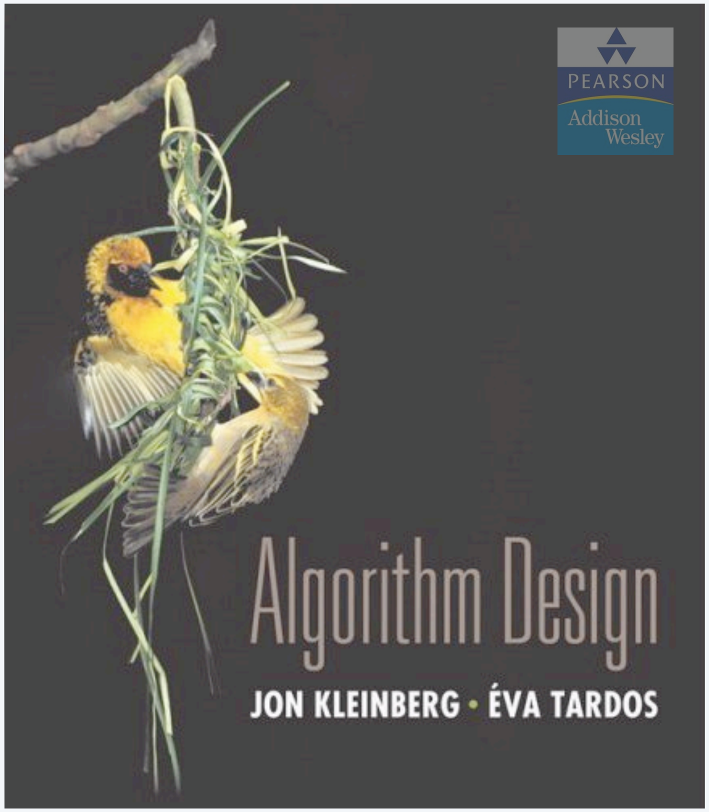
Pf.

- 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 $= X_0 + X_1 + \ldots + X_{n-1}$.

$$E[X] = \sum_{j=0}^{n-1} E[X_j] = \sum_{j=0}^{n-1} \frac{n}{n-j} = n \sum_{i=1}^{n} \frac{1}{i} = n H(n)$$

prob of success = (n – j) / n

⇒ expected waiting time = n / (n – j)



SECTION 13.9

13. RANDOMIZED ALGORITHMS

- *content resolution*
- *global min cut*
- *linearity of expectation*
- *max 3-satisfiability*
- *universal hashing*
- *Chernoff bounds*
- *load balancing*

Chernoff Bounds (above mean)

Theorem. Suppose $X_1, \ldots, X_n$ are independent 0-1 random variables. Let $X = X_1 + \ldots + X_n$. Then for any $\mu \geq E[X]$ and for any $\delta > 0$, we have

$$\Pr[X > (1+\delta)\mu] < \left[\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right]^{\mu}$$

sum of independent 0-1 random variables is tightly centered on the mean

Pf. We apply a number of simple transformations.

- For any $t > 0$,

$$\Pr[X > (1+\delta)\mu] = \Pr\left[e^{tX} > e^{t(1+\delta)\mu}\right] \leq e^{-t(1+\delta)\mu} \cdot E[e^{tX}]$$

f(x) = e^{tX} is monotone in x

Markov's inequality: $\Pr[X > a] \leq E[X] / a$

- Now $\quad E[e^{tX}] = E[e^{t \sum_i X_i}] = \prod_i E[e^{t X_i}]$

definition of X

independence

Chernoff Bounds (above mean)

Pf. [continued]

- Let $p_i = \Pr[X_i = 1]$. Then,

$$E[e^{tX_i}] \;=\; p_i e^t + (1 - p_i)e^0 \;=\; 1 + p_i(e^t - 1) \;\leq\; e^{p_i(e^t - 1)}$$

for any $\alpha \geq 0$, $1+\alpha \leq e^{\alpha}$

- Combining everything:

$$\Pr[X > (1+\delta)\mu] \;\leq\; e^{-t(1+\delta)\mu} \prod_i E[e^{tX_i}] \;\leq\; e^{-t(1+\delta)\mu} \prod_i e^{p_i(e^t-1)} \;\leq\; e^{-t(1+\delta)\mu}\; e^{\mu(e^t-1)}$$

previous slide   inequality above $\sum_i p_i = E[X] \leq \mu$

- Finally, choose $t = \ln(1 + \delta)$. ▪

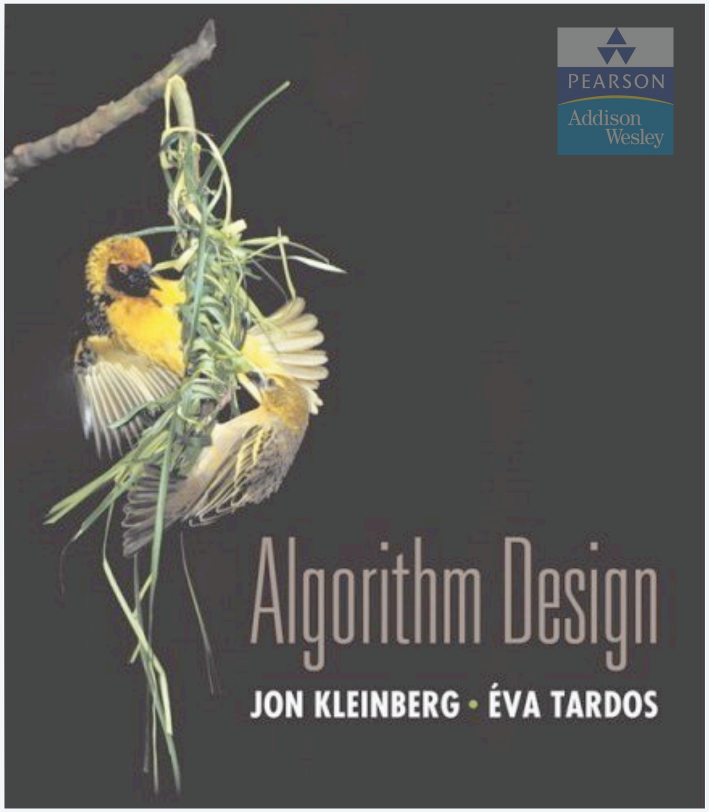
Chernoff Bounds (below mean)

Theorem. Suppose $X_1, \ldots, X_n$ are independent 0-1 random variables. Let $X = X_1 + \ldots + X_n$. Then for any $\mu \le E[X]$ and for any $0 < \delta < 1$, we have

$$\Pr[X < (1-\delta)\mu] \;<\; e^{-\delta^2 \mu / 2}$$

Pf idea. Similar.

Remark. Not quite symmetric since only makes sense to consider $\delta < 1$.



SECTION 13.10

13. RANDOMIZED ALGORITHMS

- *content resolution*
- *global min cut*
- *linearity of expectation*
- *max 3-satisfiability*
- *universal hashing*
- *Chernoff bounds*
- *load balancing*

Load Balancing

Load balancing. System in which m jobs arrive in a stream and need to be processed immediately on m identical processors. Find an assignment that balances the workload across processors.

Centralized controller. Assign jobs in round-robin manner. Each processor receives at most $\lceil m / n \rceil$ jobs.

Decentralized controller. Assign jobs to processors uniformly at random. How likely is it that some processor is assigned "too many" jobs?

Load balancing

Analysis.

- Let X_i = number of jobs assigned to processor i.
- Let $Y_{ij} = 1$ if job j assigned to processor i, and 0 otherwise.
- We have $E[Y_{ij}] = 1/n$.
- Thus, $X_i = \sum_j Y_{ij}$, and $\mu = E[X_i] = 1$.
- Applying Chernoff bounds with $\delta = c - 1$ yields $\Pr[X_i > c] < \dfrac{e^{c-1}}{c^c}$
- Let $\gamma(n)$ be number x such that $x^x = n$, and choose $c = e\,\gamma(n)$.

$$\Pr[X_i > c] < \frac{e^{c-1}}{c^c} < \left(\frac{e}{c}\right)^c = \left(\frac{1}{\gamma(n)}\right)^{e\gamma(n)} < \left(\frac{1}{\gamma(n)}\right)^{2\gamma(n)} = \frac{1}{n^2}$$

- Union bound $\Rightarrow$ with probability $\geq 1 - 1/n$ no processor receives more than $e\,\gamma(n) = \Theta(\log n / \log\log n)$ jobs.

Bonus fact: with high probability,
some processor receives Θ(logn / log log n) jobs

Load balancing: many jobs

Theorem. Suppose the number of jobs $m = 16\, n \ln n$. Then on average, each of the n processors handles $\mu = 16 \ln n$ jobs. With high probability, every processor will have between half and twice the average load.

Pf.

- Let X_i , Y_{ij} be as before.
- Applying Chernoff bounds with $\delta = 1$ yields

$$\Pr[X_i > 2\mu] < \left(\frac{e}{4}\right)^{16 n \ln n} < \left(\frac{1}{e}\right)^{\ln n} = \frac{1}{n^2} \qquad \Pr[X_i < \tfrac{1}{2}\mu] < e^{-\frac{1}{2}\left(\frac{1}{2}\right)^2 (16 n \ln n)} = \frac{1}{n^2}$$

- Union bound $\Rightarrow$ every processor has load between half and twice the average with probability $\geq 1 - 2/n$. ▪