Webpage: https://www.csa.iisc.ac.in/~arindamkhan/courses/RandAlgo23/RandAlgo23.html

E0 234: Introduction to Randomized Algorithms, Spring 2023

Teams join code: qq9jnyh

Instructors: Arindam Khan and Jaikumar Radhakrishnan
TA: Aditya Subramanian
Time: Mondays & Wednesdays, 14:00-15:30, CSA112.

Course Description | Lectures | Assignments | Projects | References

Course Description

Tentative topics: Basics of probability, Monte Carlo and Las Vegas algorithms, Karger's min-cut Algorithm, coupon collector, quicksort. Moments and Deviation: Markov's and Chebyshev's inequality, power of sampling: a randomized algorithm for computing the median. Concentration inequalities (Chernoff bounds), application. Balls and bins, birthday paradox, Poisson distribution, hashing, random graphs, threshold behavior in random graphs. Lovasz local lemma and Moser-Tardos algorithm. Introduction to algebra and probability: primality testing, verifying matrix multiplication, polynomial identity testing, randomized communication complexity, frequency moments in streams. Markov chains, random walks. Monte Carlo methods, coupling. VC dimensions, epsilon net, epsilon sample, PAC, and agnostic learning. Randomized data structures, randomized geometric algorithms.

Lectures

- Probability Refresher: Scribe notes from Toolkit, Handwritten notes.
- Lectures 1-4 (Arindam): Basics, Karger's min-cut algorithm, coupon collector, quicksort, complexity classes, Monte Carlo, and Las Vegas algorithms. [M-U Chapter 1, 2]
  Notes.
  Related Links:
  Randomized Complexity Classes (Arora-Barak),
  Different min-cut algorithms,
  Karger-Stein paper,
  STOC'21 deterministic mincut paper,
  Anupam Gupta's talk on k-cut,
  Backward analysis of quicksort.

We will be teaching materials from multiple books/sources. Some of them are the following.

- [M-U] Michael Mitzenmacher and Eli Upfal. Probability and computing. Cambridge university press, 2017.
- [MR] Rajeev Motwani, Prabhakar Raghavan. Randomized Algorithms, Cambridge university press.
- [RK] R.M. Karp, An introduction to randomized algorithms, Discrete Applied Mathematics, 34, pp. 165-201, 1991.
- [BHK] Avrim Blum, John Hopcroft, and Ravindran Kannan. Foundations of Data Science, 2020.
- [RV] Roman Vershynin, High-Dimensional Probability.
- [DP] D.B. Dubhashi, A. Panconesi, Concentration of Measure for the Analysis of Randomized Algorithms, Cambridge University Press, 2009.
- [LPW] David A. Levin, Yuval Peres, Elizabeth L. Wilmer. Markov Chains and Mixing Times.
- [MIT-YZ] Yufei Zhao. Lecture Notes (The Probabilistic Methods in Combinatorics), MIT, 2019.
- [A-S] Noga Alon and Joel Spencer, The probabilistic method, John Wiley & Sons, 2004.
- [UW-TR] Thomas Rothvoss, Lecture Notes (Probabilistic Combinatorics), U Washington, 2019.
- [AC] Amit Chakrabarti, Data Stream Algorithms, 2020.
- [SM] S. Muthukrishnan. Data streams: Algorithms and applications. Now Publishers Inc, 2005.
- Various surveys and lecture notes.

Similar courses elsewhere:
- IISc, 2021 , by Siddharth Barman and Arindam Khan.] past offering
- Yale, 2020 , by James Aspnes.
- IISc, 2016 , by Arnab Bhattacharyya and Deeparnab Chakrabarty.
- Yale, 2020 , by James Aspnes.
- [SH-UIUC] UIUC, 2018 , by Sariel Har-Peled.] detailed lecture notes
- MIT, 2002 , by David Karger.
- UT Austin, 2020 , by Eric Price.
- UC berkeley, 2003 , by Luca Trevisan.
- Columbia, 2019 , by Tim Roughgarden.
- Stanford, 2020 , by Mary Wooters.
- CMU, 1997 , by Avrim Blum.
- Wiezmann, 2013 , by Robert Krauthgamer and Moni Naor.
- UMCP, 2017 , by Aravind Srinivasan.
- U Iowa, 2018 , by Sriram V. Pemmaraju.
- UBC, 2012 , by Nick Harvey.
- EPFL 2014 , by Friedrich Eisenbrand.
- NUS, 2019 , by Seth Gilbert.
- Duke, 2013 , by Kamesh Munagala.
- NTHU, 2012 , by Wing Kai Hon.
- U Waterloo, 2019, by Gautam Kamath.
- U Waterloo, 2018 , by Lap Chi Lau.
- U Washington, 2016 , by James Lee.

Fun links:
- Drunkard's Walk, Book by Leonard Mlodinow.
- Veritasium Video, How We are Fooled By Probability: Regression to the Mean.
- Vsauce Video, Birthday Paradox.
- Numberphile Video, Monty Hall Problem.
- Sunlight is way older than you think, An interesting application of Random Walks (Markov chains).
- Veritasium Video, The Bayesian Trap.
- MindYourDecisions Video, Buffon's Needle Problem: Pi from Probability.

Assignments

→ Latexed Soln.

Project Topics

For projects you need to select a project topic (to be announced later). Some reference papers will be given. You are expected to do a survey of the results and techniques in the topic area. You can form a group of two students and send the project topic and group details by 20th February. You need to submit a report by the last_day_of_class. Finally, you need to make a presentation on the topic on the last week of classes.

Intended audience: Graduate students in computer science and mathematics with theoretical interests. Interested undergraduate students with very strong mathematical skills.

Acad integrity

Prerequisites: Mathematical maturity and a solid background in math (elementary combinatorics, graph theory, discrete probability, algebra, calculus) and theoretical computer science (big-O/Omega/Theta, P/NP, basic fundamental algorithms).

Grading: 40% HW, 30% Projects, 30% Final.

- Randomized Algorithm.

§ What is randomness?

→ lack of predictability / certainty.

e.g. coin toss: given a sequence we cannot predict the next outcome with certainty.

Dubai
D&N

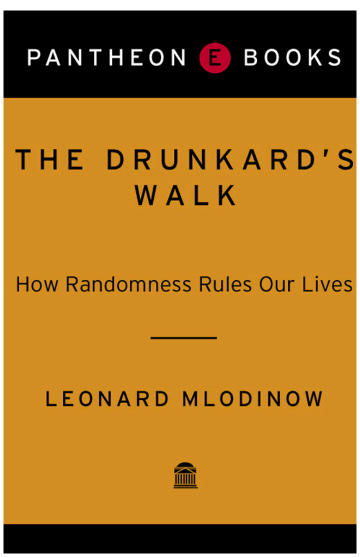
Player	Span	Matches	Toss won	Percentage
Ricky Ponting	2002-2012	324	170	52.47
Graeme Smith	2003-2014	286	148	51.75
Stephen Fleming	1997-2007	303	147	48.51
Allan Border	1984-1994	271	132	48.70
Arjuna Ranatunga	1988-1999	249	132	53.01
Mohd. Azharuddin	1990-1999	221	125	56.56
MS Dhoni	2007-2014	256	123	48.05
Hansie Cronje	1994-2000	191	95	49.74
Sourav Ganguly	1999-2005	196	95	48.47
Imran Khan	1982-1992	187	95	50.80

- Game of dice in Mahabharat.
  Snake & ladder (Mokshapatnam).

Why didn't Greeks invent probability?

- First systematic study on probability:
  Liber de ludo aleae (Book on games of chance)
  - by Gerolamo Cardano (≈ 1564)

§ Randomization in life:



'God does not play dice with the universe'
- However, randomness has become an essential component in understanding, modeling, and analyzing nature.

→ Subparticle physics: governed by random behavior & statistical laws,
- Brownian motion, radioactive decay ...

→ Biology: mutation & evolution.

→ Economics: price fluctuations in free-market economy.

↳ Story of best performing Mutual funds are by dead people!

PROBABILITY REFRESHER: [Source: Mitzenmacher-Upfal]

Definition 1.1: *A probability space has three components:*

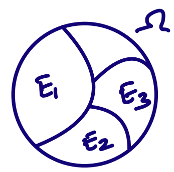
1. *a sample space* Ω, *which is the set of all possible outcomes of the random process modeled by the probability space;* → $\Omega := \{H, T\}$
2. *a family of sets* $\mathcal{F}$ *representing the allowable events, where each set in* $\mathcal{F}$ *is a subset*[1] *of the sample space* Ω*; and* → $2^\Omega = \{\phi, \Omega, \{H\}, \{T\}\}$ (σ-field, set of events)
3. *a probability function* $\Pr : \mathcal{F} \to \mathbf{R}$ *satisfying Definition 1.2.*

An element of Ω is called a *simple* or *elementary* event.

Definition 1.2: *A probability function is any function* $\Pr : \mathcal{F} \to \mathbf{R}$ *that satisfies the following conditions:*

1. *for any event* E, $0 \le \Pr(E) \le 1$;
2. $\Pr(\Omega) = 1$; *and*
3. *for any finite or countably infinite sequence of pairwise mutually disjoint events* $E_1, E_2, E_3, \ldots,$

$$\Pr\left(\bigcup_{i \ge 1} E_i\right) = \sum_{i \ge 1} \Pr(E_i).$$



[A collection $\mathcal{F}$ of subsets of Ω is called σ-field if:

(i) $\Omega \in \mathcal{F}$

(ii) $A \in \mathcal{F} \Rightarrow A^c \in \mathcal{F}$

(iii) $A_1, A_2, \ldots, A_n \in \mathcal{F} \Rightarrow \bigcup_{i=1}^{n} A \in \mathcal{F}$

→ In this course we will use discrete probability space, i.e. sample space Ω is finite or countably infinite, and the family $\mathcal{F}$ of allowable events consists of all subsets of Ω [$\mathcal{F} = 2^\Omega$].

→ In a discrete prob space, probability function is uniquely defined by probabilities of simple events.

- Events are sets.

say, we roll two dice.

E_1 is event first die is 6 & E_2 is event second die is 6.

Then think about events $E_1 \cup E_2$, $E_1 - E_2$, $E_1 \cap E_2$, $\Omega - E$.

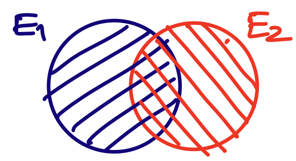
Lemma 1.2: *For any finite or countably infinite sequence of events $E_1, E_2, \ldots,$*

↳ may not be pairwise disjoint

∘ Union bound:

$$\Pr\left(\bigcup_{i\geq 1} E_i\right) \leq \sum_{i\geq 1} \Pr(E_i).$$

∘ Inclusion-exclusion principle:



Lemma 1.3: *Let $E_1, \ldots, E_n$ be any n events. Then*

$$\Pr\left(\bigcup_{i=1}^{n} E_i\right) = \sum_{i=1}^{n} \Pr(E_i) - \sum_{i<j} \Pr(E_i \cap E_j) + \sum_{i<j<k} \Pr(E_i \cap E_j \cap E_k)$$

$$- \cdots + (-1)^{\ell+1} \sum_{i_1<i_2<\cdots<i_\ell} \Pr\left(\bigcap_{r=1}^{\ell} E_{i_r}\right) + \cdots.$$

Definition 1.3: *Two events E and F are independent if and only if*

$$\Pr(E \cap F) = \Pr(E) \cdot \Pr(F).$$

pairwise independence.

More generally, events $E_1, E_2, \ldots, E_k$ are mutually independent if and only if, for any subset $I \subseteq [1, k]$,

$$\Pr\left(\bigcap_{i\in I} E_i\right) = \prod_{i\in I} \Pr(E_i).$$

K-wise independence: A set of events $E_1, E_2, \ldots, E_n$ is K-wise independent if, for any subset $I \subseteq [1, n]$ with $|I| \leq K$,

$$\mathbb{P}\left[\bigcap_{i\in I} E_i\right] = \prod_{i\in I} \mathbb{P}[E_i]$$

Definition 1.4: *The conditional probability that event E occurs given that event F occurs is*

$$\Pr(E \mid F) = \frac{\Pr(E \cap F)}{\Pr(F)}.$$

The conditional probability is well-defined only if $\Pr(F) > 0$.

Chain rule: $\Pr\left(\bigcap_{i=1}^{k} \mathcal{E}_i\right) = \prod_{i=1}^{k} \Pr\left(\mathcal{E}_i \mid \bigcap_{j=1}^{i-1} \mathcal{E}_j\right)$

$= \Pr(\mathcal{E}_1).\Pr(\mathcal{E}_2 \mid \mathcal{E}_1).\Pr(\mathcal{E}_3 \mid \mathcal{E}_1 \cap \mathcal{E}_2) \cdots \Pr(\mathcal{E}_k \mid \cap_{i=1}^{k-1} \mathcal{E}_i).$

- Repeatedly choosing random numbers acc. to a given distribution → sampling.
- with replacement → simple to code
  without replacement → gives slightly better bounds.

Theorem 1.6 [Law of Total Probability]: *Let $E_1, E_2, \ldots, E_n$ be mutually disjoint events in the sample space Ω, and let $\bigcup_{i=1}^{n} E_i = \Omega$. Then*

$$\Pr(B) = \sum_{i=1}^{n} \Pr(B \cap E_i) = \sum_{i=1}^{n} \Pr(B \mid E_i)\Pr(E_i).$$

Theorem 1.7 [Bayes' Law]: *Assume that $E_1, E_2, \ldots, E_n$ are mutually disjoint events in the sample space Ω such that $\bigcup_{i=1}^{n} E_i = \Omega$. Then*

$$\Pr(E_j \mid B) = \frac{\Pr(E_j \cap B)}{\Pr(B)} = \frac{\Pr(B \mid E_j)\Pr(E_j)}{\sum_{i=1}^{n} \Pr(B \mid E_i)\Pr(E_i)}.$$

∵ LTP

$$= \frac{\mathbb{P}(B \mid E_j)\,\mathbb{P}(E_j)}{\mathbb{P}[B]}.$$

∘ Randomness is counter-intuitive:

Daniel Kahneman (2002 Economics Nobel) and Tversky in Prospect theory established a cognitive basis for human errors that arise from heuristics & biases.

(I) Airplane manouvers & regression towards the mean.

" An extraordinary event is more likely to be followed by an ordinary ones."

[Source: Kahneman & Israeli flight instructors from The Drunkard's Walk]

(II) Buying lottery & flying airplanes:

Air travel resulted in 0.07 deaths for every 1 billion miles travelled compared to 212.57 for motorcycles and 7.28 for cars. We will continue to make the skies safer and you continue to fly!

(III) My dad heard this story on the radio. At Duke University, two students had received A's in chemistry all semester. But on the night before the final exam, they were partying in another state and didn't get back to Duke until it was over. Their excuse to the professor was that they had a flat tire, and they asked if they could take a make-up test. The professor agreed, wrote out a test, and sent the two to separate rooms to take it. The first question (on one side of the paper) was worth five points. Then they flipped the paper over and found the second question, worth 95 points: "which tire was it?" What was the probability that both students would say the same thing? My dad and I think it's 1 in 16. Is that right?[14]

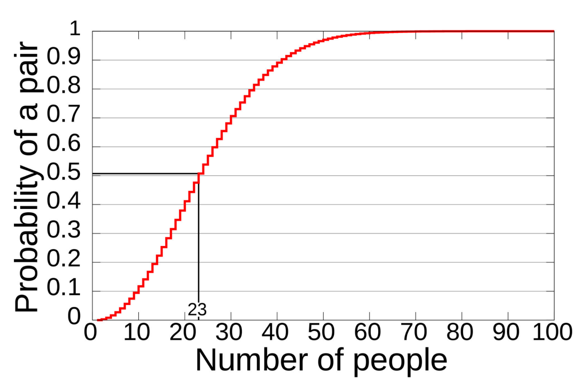
'In no other branch of mathematics it is so easy for experts to blunder as in probability theory'

— Martin Gardner.

Independence!

(III) Canadian lottery & Birthday paradox.

A few years ago Canadian lottery officials learned the importance of careful counting the hard way when they decided to give back some unclaimed prize money that had accumulated.[3] They purchased 500 automobiles as bonus prizes and programmed a computer to determine the winners by randomly selecting 500 numbers from their list of 2.4 million subscriber numbers. The officials published the unsorted list of 500 winning numbers, promising an automobile for each number listed. To their embarrassment, one individual claimed (rightly) that he had won two cars. The officials were flabbergasted—with over 2 million numbers to choose from, how could the computer have randomly chosen the same number twice? Was there a fault in their program?



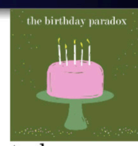
31 days in a month.

For randomly selected 7 people, they dont share same birthdate

$= \frac{31}{31} \times \frac{30}{31} \times \frac{29}{31} \times \frac{28}{31} \times \frac{27}{31} \times \frac{26}{31} \times \frac{25}{31} \approx 0.48 \Rightarrow \mathbb{P}(\text{Success}) \approx 0.52$

For 10 people $\approx 0.196. \Rightarrow \mathbb{P}(\text{Success}) \approx 0.804.$

The Birthday Paradox



- *What is the probability $p(n)$ such that in a set of n randomly chosen people, two people will have the same birthday?*
- The probability $\overline{p}(n) = 1 - p(n)$ can be easily calculated:

$$\overline{p}(n) = \frac{365 \times 364 \times \cdots \times 365 - n + 1}{365^n} = \prod_{K=0}^{n-1}\left(1 - \frac{K}{365}\right) = e^{\sum_{K=0}^{n-1} \log\left(1 - \frac{K}{365}\right)}$$
$$\approx e^{-\sum_{K=0}^{n-1} \frac{K}{365}}$$
$$= e^{-\frac{n(n-1)}{2 \times 365}}.$$

- *The minimum n such that $p(n) \geq \frac{1}{2}$ is about* $\sqrt{2 \times 365 \times \log 2} \approx 23$.

Birthday attack

Pollard's rho algorithm: algorithm for integer factorization

Birthday Coincidences

Question

What is the chance that there are two people in the US who (a) know each other, (b) have the same birthday, (c) their fathers have the same birthday, (d) their grandfathers have the same birthday, and (e) their great grandfathers have the same birthdays.

Excluding siblings! or cousins.

- Estimated number of edges in the US friendship graph G_n is about $|E(G_n)| = \frac{1}{2} \times 600 \times 400 \times 10^6$.
- The 4-fold birthday coincidence amounts to $c_n = (365)^4$ 'colors'.
- Then, by the Poisson approximation result, the chance of a match is

$$\mathbb{P}(T(K_2, G_n) > 0) \approx 1 - e^{-|E(G_n)|/c} \approx 99.8\%. \qquad \text{(CRAZY!!!)}$$

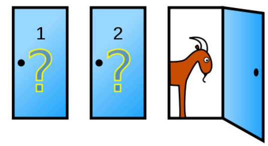
- By the $\mathbb{P}(T(K_2, G_n) > 0) \geq 1 - \frac{1}{|E(G_n)|/c} \approx 85\%$ (STILL CRAZY!!!).
- The phenomenon of *the law of truly large numbers* (Diaconis and Mosteller (1989)): *when enormous numbers of events and people and their interactions cumulate over time, almost any outrageous event is bound to occur.*

Slides: Bhaswar B.

(IV)

The **Monty Hall problem** is a brain teaser, in the form of a probability puzzle, loosely based on the American television game show *Let's Make a Deal* and named after its original host, Monty Hall. The problem was originally posed (and solved) in a letter by Steve Selvin to the *American Statistician* in 1975.[1][2] It became famous as a question from a reader's letter quoted in Marilyn vos Savant's "Ask Marilyn" column in *Parade* magazine in 1990:[3]

> Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you, "Do you want to pick door No. 2?" Is it to your advantage to switch your choice?



In search of a new car, the player picks a door, say *1*. The game host then opens one of the other doors, say *3*, to reveal a goat and offers to let the player switch from door *1* to door *2*.

Vos Savant's response was that the contestant should switch to the other door.[3] Under the standard assumptions, contestants who switch have a $\frac{2}{3}$ chance of winning the car, while contestants who stick to their initial choice have only a $\frac{1}{3}$ chance.

Assumptions:

1. The host must always open a door that was not picked by the contestant.[9]
2. The host must always open a door to reveal a goat and never the car.
3. The host must always offer the chance to switch between the originally chosen door and the remaining closed door.

You fix Door 1.

Bayes' theorem: $\mathbb{P}(A \mid B) = \mathbb{P}(B \mid A) \cdot \mathbb{P}(A) / \mathbb{P}(B)$.

A: flash car behind door 1
B: Monty opened door 3 & show goat

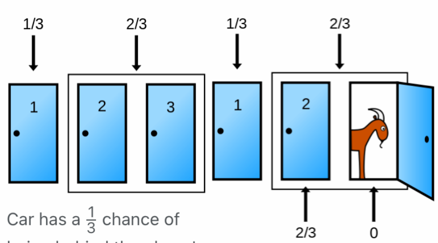
$\mathbb{P}[\text{Monty opens door 3 \& show goat} \mid \text{car is behind door 1}] = \frac{1}{2}$

$\mathbb{P}[A] = \frac{1}{3}$, $\mathbb{P}[B] = \frac{1}{2}$ → By symmetry.

$\mathbb{P}(A \mid B) = \frac{1}{2} \cdot \frac{1}{3} / \frac{1}{2} = \frac{1}{3}$.

switch

$\Rightarrow \mathbb{P}(\text{flash car behind door 2} \mid \text{Monty opens door 3 \& show goat}) = \frac{2}{3}$.

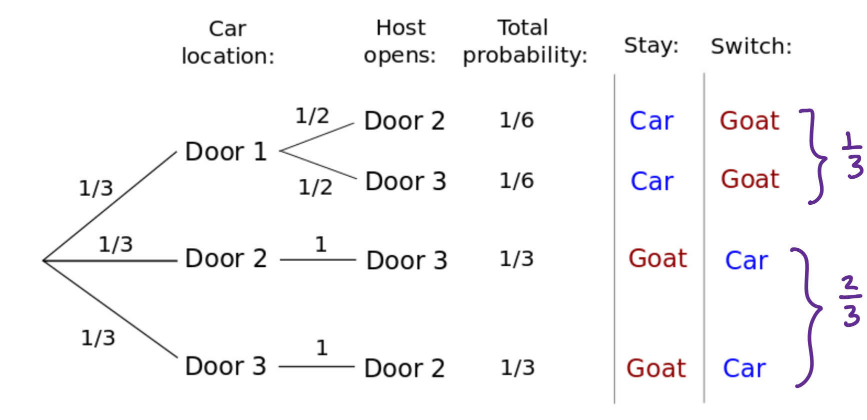


Car has a $\frac{1}{3}$ chance of being behind the player's pick and a $\frac{2}{3}$ chance of being behind one of the other two doors.

The host opens a door, the odds for the two sets don't change but the odds move to 0 for the open door and $\frac{2}{3}$ for the closed door.

Assume there are 10^6 doors.
$10^6 - 1$ has goats & 1 car.
Player picks a door & host opens $10^6 - 2$ doors w. goat.
Will u switch?

image source: wikipedia



ⓥ Simpson's paradox.

Simpson's paradox is a phenomenon in probability and statistics in which a trend appears in several groups of data but disappears or reverses when the groups are combined. This result is often encountered in social-science and medical-

Batting averages

A common example of Simpson's paradox involves the batting averages of players in professional baseball. It is possible for one player to have a higher batting average than another player each year for a number of years, but to have a lower batting average across all of those years. This phenomenon can occur when there are large differences in the number of at bats between the years. Mathematician Ken Ross demonstrated this using the batting average of two baseball players, Derek Jeter and David Justice, during the years 1995 and 1996:[18][19]

Batter \ Year	1995		1996		Combined	
Derek Jeter	12/48	.250	183/582	.314	195/630	**.310**
David Justice	104/411	**.253**	45/140	**.321**	149/551	.270

It is possible to have:

$\mathbb{P}(A \mid B \cap C) > \mathbb{P}(A \mid B^c \cap C)$

& $\mathbb{P}(A \mid B \cap C^c) > \mathbb{P}(A \mid B^c \cap C^c)$

but $\mathbb{P}(A \mid B) < \mathbb{P}(A \mid B^c)$

	Women: Drug 1 ✓✓	Women: Drug 2	Men: Drug 1 ✓✓	Men: Drug 2
Success	200	10	19	1000
Failure	1800	190	1	1000

Drug 1 219/2020

Drug 2 1010/2200. ✓✓

A = success

$B = D1, B^c = D2$

$C = M, C^c = F.$

§ Discrete Random Variables and Expectation. [Ch 2 M-U]

Defn: R.V. A random variable X is a function $X : \Omega \to \mathbb{R}$. A discrete R.V. is a R.V. that takes finite or countably infinite number of values.

Defn: (Expectation) $\mathbb{E}[X] = \sum_i i \cdot \mathbb{P}[X = i]$.

⊙ Properties of expectation:

Thm 2.1: [Linearity of Expectations]
For any finite collection of discrete RVs $X_1, X_2, \dots, X_n$ with finite expectations,

$$\mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i].$$

(Note: We don't need these RVs to be independent)

- Linearity of expectations also hold for countably infinite summations in certain cases:

$$\mathbb{E}\left[\sum_{i=1}^{\infty} X_i\right] = \sum_{i=1}^{\infty} \mathbb{E}[X_i] \text{ whenever } \sum_{i=1}^{\infty} \mathbb{E}[|X_i|] \text{ converges.}$$

> Consider a series $a_1, a_2, \dots$.
> $S = \sum_{k=1}^{\infty} a_k$. $S_n := \sum_{k=1}^{n} a_k$
> A series converges if there exists a number ℓ s.t. $\forall$ arbitrarily small $\varepsilon > 0$, $\exists$ a sufficiently large $N \in \mathbb{N}$ s.t. $\forall n \geq N$, $|S_n - \ell| < \varepsilon$.

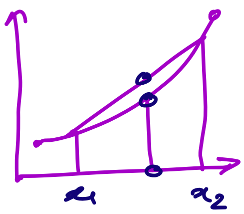
Note, convergence of $\sum a_i$ not necessarily mean convergence of $\sum |a_i|$.

e.g. $1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \dots = \ln(2)$ convergent

but $1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots = \infty$ divergent

Lem 2.2: For any constant c and discrete RV X, $\boxed{\mathbb{E}[cX] = c\,\mathbb{E}[X].}$

Defn 2.4: (convex function) A function $f: \mathbb{R} \to \mathbb{R}$ is said to be convex if, for any x_1, x_2 and $0 \leq \lambda \leq 1$, $f(\lambda x_1 + (1-\lambda) x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$.

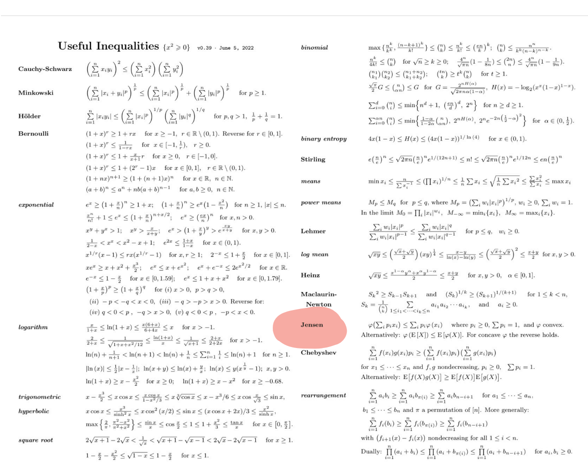


lem 2.3: If f is twice differentiable function then f is convex iff $f''(x) \geq 0$.

e.g. $f(x) = x^2, x^4, x$. e^x, $|x|^p$ for $p \geq 1$. | x^3 is partly convex.

Thm 2.4 (Jensen's Inequality) If f is a convex function then $\boxed{\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])}$

Corollary: $\mathbb{E}[X^2] \geq (\mathbb{E}[X])^2$.


Useful Inequalities $\{x^2 \geq 0\}$ v0.39 · June 5, 2022

Cauchy-Schwarz, Minkowski, Hölder, Bernoulli, exponential, logarithm, trigonometric, hyperbolic, square root, binomial, binary entropy, Stirling, means, power means, Lehmer, log mean, Heinz, Maclaurin-Newton, Jensen, Chebyshev, rearrangement

Check https://www.lkozma.net/inequalities_cheat_sheet/ineq.pdf for useful inequalities.

- ## Indicator Random Variables :

Remember random variable X is a fn $X : \Omega \to \mathbb{R}$.
e.g., we can have $X(\omega) = 1$ if $\omega \in \{2, 4, 6\}$
$= -1$ if $\omega \in \{3, 5\}$
$= -0.5$ if $\omega \in \{1\}$

Indicator random variable (or indicator function) is a random variable that can take only two values :

- → 1 if event E happens
- → 0 otherwise.

Indicator function of an event E is denoted by $\mathbb{1}_E$. also $\mathbb{I}_E$

Formally, event $E \subseteq \Omega$.

$$\mathbb{1}_E(\omega) = 1 \text{ if } \omega \in E$$
$$\mathbb{1}_E(\omega) = 0 \text{ if } \omega \notin E$$

Event : Get even number in a die roll.

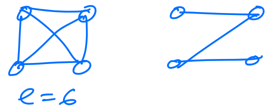
$$\mathbb{1}_E(\omega) = 1 \text{ if } \omega \in \{2, 4, 6\}$$
$$\mathbb{1}_E(\omega) = 0 \text{ if } \omega \in \{1, 3, 5\}$$

Expectation:

$$\mathbb{E}[\mathbb{1}_E] = \mathbb{P}[E] \cdot 1 + \mathbb{P}[E^c] \cdot 0 = \mathbb{P}[E].$$

HW : Prove inclusion-exclusion principle using indicator functions.

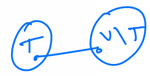
- Applications of linearity of expectations!



> Let $G = (V, E)$ be a graph with n vertices and e edges. Then it contains a bipartite subgraph of at least $\frac{e}{2}$ edges.

Probabilistic Methods!

Let $T \subseteq V$ be a random subset given by $\mathbb{P}[x \in T] = \frac{1}{2}$. An edge $\{x, y\}$ is a crossing if exactly one of x, y are in T.



we put each vertex independently.

Denote by X_{xy} the random indicator for x, y being a crossing, and set

$$X = \sum_{\{x,y\} \in E} X_{xy}.$$

pairwise independence needed here

Since $\mathbb{E}[X_{xy}] = \frac{1}{2}$, by *linearity of expectation* we have

$$\mathbb{E}[X] = \sum_{\{x,y\} \in E} \mathbb{E}[X_{xy}] = \frac{e}{2}.$$

why do we choose $\frac{1}{2}$? try p & then optimize. then we'll get $\mathbb{E}[X_{xy}] = p(1-p)$.

Consequently, there exists a choice of T with at least $\frac{e}{2}$ crossing edges, which form a bipartite graph.

HW: Can we get a polynomial-time deterministic algorithm? [Hint: Max-cut].

HW: Prove that there is a partition $V_1 \cup V_2 = V$ of vertices s.t. $\forall v \in V_1$, $|Nbrs(v) \cap V_1| \leq |Nbrs(v) \cap V_2|$ and $\forall v \in V_2$, $|Nbrs(v) \cap V_2| \leq |Nbrs(v) \cap V_1|$.

HW: Better sampling:
If G has $2n$ vertices & m edges then it contains a bipartite subgraph with $\geq mn/(2n-1)$ edges.

⊙ A uniformly random permutation $\pi : [n] \to [n]$ is expected to have a unique fixed point:

1 2 3	3
1 3 2	1
2 1 3	1
2 3 1	0
3 1 2	0
3 2 1	1

$X = \#$ fixed points in such π.

$$X = \sum_{k=1}^{n} \mathbb{1}_{\pi(k)=k}$$

Indicator Random Variable (IRV): A random variable that has value 1 or 0, acc. to whether a specified event E occurs or not is called IRV for E.

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{k=1}^{n} \mathbb{1}_{\pi(k)=k}\right]$$

lin. of expectation

$$= \sum_{k=1}^{n} \mathbb{E}\left[\mathbb{1}_{\pi(k)=k}\right]$$

$$= \sum_{k=1}^{n} \mathbb{P}(\pi(k)=k) = \sum_{k=1}^{n} \frac{1}{n} = 1.$$

HW: # fixed points converges to a Poisson distribution. More specifically, $\mathbb{P}[X=j]$ converges to $\frac{1}{e \cdot (j!)}$.

Buffon's needle: rule a surface with parallel lines a distance d apart. What is the probability that a randomly dropped needle of length $\ell \leq d$ crosses a line?

Consider dropping *any* (continuous) curve of length ℓ onto the surface. Imagine dividing up the curve into N straight line segments, each of length ℓ/N. Let X_i be the indicator for the i-th segment crossing a line. Then if X is the total number of times the curve crosses a line,

$$\mathbb{E}[X] = \mathbb{E}\left[\sum X_i\right] = \sum \mathbb{E}[X_i] = N \cdot \mathbb{E}[X_1].$$

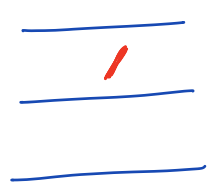
That is to say, the expected number of crossings is proportional to the length of the curve (and independent of the shape).

Now we need to fix the constant of proportionality. Take the curve to be a circle of diameter d. Almost surely, this curve will cross a line twice. The length of the circle is πd, so a curve of length ℓ crosses a line $\frac{2\ell}{\pi d}$ times.

Now observe that a straight needle of length $\ell \leq d$ can cross a line either 0 or 1 times. So the probability it crosses a line is precisely this expectation value $\frac{2\ell}{\pi d}$.

Estimate π

$(\ell = 1, d = 2)$



§ Bernoulli, Binomial and Geometric RV :

Bernoulli : Y be a RV s.t. $Y = \begin{cases} 1 & \text{w.p. } p \\ 0 & \text{w.p. } (1-p). \end{cases}$

$\mathbb{E}[Y] = p.1 + (1-p).0 = p.$

e.g. one coin toss can be modeled by Bernoulli.

- Indicator random variables are related.

Binomial : $X := \text{Bin}(n,p)$ is a random variable taking the values $0, 1, 2, \dots, n$ and

$\mathbb{P}(X=k) = \binom{n}{k} p^k (1-p)^{n-k}$ where $0<p<1$.

e.g. n coin toss. How many heads?

useful in sampling. $\mathbb{E}[X] = \mathbb{E}[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} \mathbb{E}[X_i] = np.$

Geometric : $X := \text{Geom}(p)$ is a geometric RV if X takes values $1, 2, 3, \dots$ with $\mathbb{P}(X=k) = p(1-p)^{k-1}$.

e.g. number of coin flips till you get a first head.

Theorem : $\mathbb{E}[X] = 1/p$, for $X := \text{Geom}(p)$.

Define

$X_i := \begin{cases} 1 & \text{if at least } i \text{ trials are needed for success.} \\ 0 & \text{otherwise} \end{cases}$

Then, we have $X = \sum_{i=1}^{\infty} X_i$.

$$\Rightarrow \mathbb{E}[X] = \sum_{i\geq 1} \mathbb{E}[X_i] = \sum_{i\geq 1} (1-p)^{i-1} = \sum_{j\geq 0} (1-p)^j = \frac{1}{1-(1-p)} = \frac{1}{p}.$$

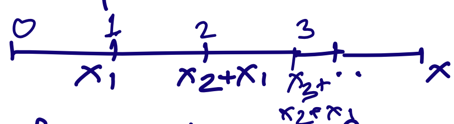
Example: Coupon Collector's Problem.

Suppose each box of cereal contains one of n different coupons. Once you obtain one of every type of coupon, you can send in for a prize. Assuming coupon in each box is chosen indepen--dently and uniformly at random, how many boxes of cereal you need to buy before you obtain at least one of every type of coupon.

- Let X be #boxes bought until we have all types of coupons.

  Let X_i denote # new boxes bought while you had exactly $(i-1)$ different coupons, then clearly

  $$X = \sum_{i=1}^{n} X_i.$$



  When exactly $(i-1)$ coupons have been found, the prob. of obtaining a new coupon is:

  $$P_i = 1 - \frac{i-1}{n}.$$

  Hence, X_i is a geom RV with parameter p_i.

  $$\mathbb{E}[X_i] = \frac{1}{p_i} = \frac{n}{n-i+1}.$$

  * with replacement

  $$\therefore \mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}[X_i]$$

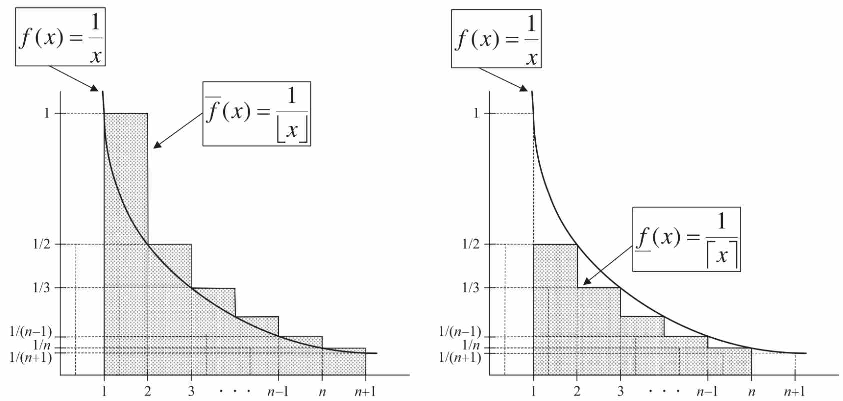
  $$= \sum_{i=1}^{n} \frac{n}{n-i+1} = n \sum_{j=1}^{n} \frac{1}{j} = n\,H(n).$$

  where $H(n)$ is Harmonic number $= \ln n + \Theta(1)$.

Note: $\sum \frac{1}{n^c} \to \infty$ as $n \to \infty$ if $c \le 1$, else it is finite. $\sum \frac{1}{n^2} = \pi^2/6$.

[HW: prove: $H(n) = \ln(n) + \Theta(1)$.

Hint:



(a) Approximating $1/x$ from above. (b) Approximating $1/x$ from below.

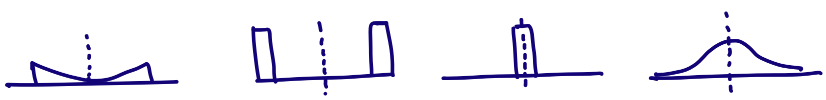
Figure 2.1: Approximating the area above and below $f(x) = 1/x$.

]

- Is X concentrated?

→ Expectation tells mean.

But underlying distributions could be quite different.



Later we'll see the following bound using some advanced methods such as Chernoff bounds & Poisson approximation.

Theorem 5.13: *Let X be the number of coupons observed before obtaining one of each of n types of coupons. Then, for any constant c,*

$$\lim_{n\to\infty} \Pr[X > n\ln n + cn] = 1 - e^{-e^{-c}}.$$

This theorem states that, for large n, the number of coupons required should be very close to $n \ln n$. For example, over 98% of the time the number of coupons required lies between $n \ln n - 4n$ and $n \ln n + 4n$. This is an example of a *sharp threshold*, where the random variable is closely concentrated around its mean.

§ Importance in Computer Science

- Randomized algorithms are algorithms that make random choices during their execution.
- Advantage: simplicity, speed.

Random number generation is a process which, often by means of a **random number generator** (**RNG**), generates a sequence of numbers or symbols that cannot be reasonably predicted better than by a random chance. Random number generators can be truly random *hardware random-number generators* (HRNGS), which generate random numbers as a function of current value of some physical environment attribute that is constantly changing in a manner that is practically impossible to model, or pseudorandom number generators (PRNGS), which generate numbers that look random, but are actually deterministic, and can be reproduced if the state of the PRNG is known.

Example: Given an integer array of size $2K+1$, return an element $\geq$ median of the array.

→ Best deterministic algo need to read $(K+1)$ elements.

→ Randomized algo: Choose 100 numbers randomly, Return the max.

(randomly → uniformly at random, each time independently.)

Success probability?

For one number: $\frac{K+1}{2K+1} \approx \frac{1}{2}$.

For 100 numbers: $\left(\frac{1}{2}\right)^{100} \to 0$. (surprisingly independent of K.)

Here, we use sampling with replacements.

It is easier to analyze, though sampling with replacement might give better bounds.

- Power of randomness: Beating worst case performance of a single deterministic algo.

or performance guarantee

① **Foiling an adversary**: A lower bound on the running time of a deterministic algorithm comes from an input on which the algorithm fares poorly. Thus the worst-case input can be different for diff. algorithms.

$A_1 - I_1$ ✓
$A_2 - I_2$ ✓
$A_3 - I_3$ ✓

One can interpret this as an adversary choosing the worst-case for a given algorithm.

A randomized algorithm can be viewed as a probability distribution on a set of deterministic algorithm.

Adversary may devise an input that foils one (or a small fraction) of the deterministic algorithms, it is difficult to construct a single input that is likely to defeat a randomly chosen algorithm.

Similarly sometimes random reordering of input data followed by application of relatively naive algorithms, is very powerful.
(Quicksort example: either random pivot or random ordering of input)

② **Random sampling**: A random sample from a population is representative of the population as a whole.

(Random survey)
RCT

③ **Probabilistic methods:** We show a randomly chosen object has some property with positive probability, then such objects exists and we can then find them using constructive efficient algorithm.

④ **Other applications:** Hashing, Monte Carlo simulations & primality testing, ...

§ **ML & related areas:** ML & Data mining create, collect & store massive data sets. Randomness helps in modeling, under-standing, and making predictions based on large data sets, and relates accuracy & sample size.
[sample complexity, VC dimension, Rademacher averages]

§ **Probabilistic analysis:** Sometimes hard to compute problems are often easy in practice. Prob. analysis sometimes give a theoretical explanation of this phenomena. In this case we assume the input to be randomly selected acc. to some prob distr. on the collection of all possible inputs, and might provide efficient algorithms on almost all inputs.

We will focus on rand. algo, not prob analysis.

⊙ Minimum Cut Problem. [Global mincut]

In the mid-1950s, U. S. Air Force researcher Theodore E. Harris and retired U. S. Army general Frank S. Ross wrote a classified report studying the rail network that linked the Soviet Union to its satellite countries in Eastern Europe. The network was modeled as a graph with 44 vertices, representing geographic regions, and 105 edges, representing links between those regions in the rail network. Each edge was given a weight, representing the rate at which material could be shipped from one region to the next. Essentially by trial and error, they determined both the maximum amount of stuff that could be moved from Russia into Europe, as well as the cheapest way to disrupt the network by removing links (or in less abstract terms, blowing up train tracks), which they called "the bottleneck". Their report, which included the drawing of the network in Figure 10.1, was only declassified in 1999.[1]

Source: Algorithms textbook by Jeff Erickson

[1] I learned this story from Alexander Schrijver's fascinating survey "On the history of combinatorial optimization (till 1960)"; the Harris-Ross report was declassified at Schrijver's request. Ford and Fulkerson (who we will meet shortly) credit Harris for formulating the problem. They thank Dantzig.

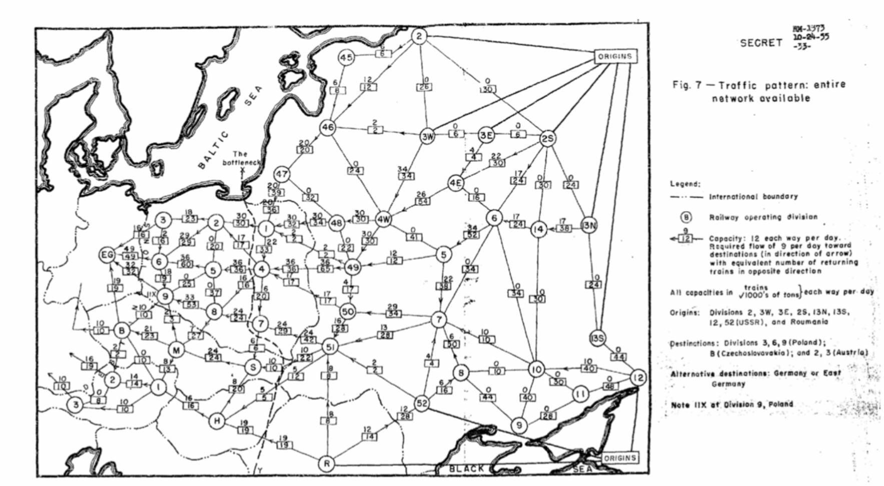


Figure 10.1. Harris and Ross's map of the Warsaw Pact rail network. (See Image Credits at the end of the book.)

Given: $G := (V, E)$ connected, undirected multigraph; $|V| = n$.

Goal: Find a min-cut. (cut of min cardinality)

where a cut is a set of edges whose removal results in G being broken into two or more components.

Option 1 : via s-t min-cut via max-flow
[celebrated max-flow min cut]

Gomory-Hu [1961] : Compute min cut via (n-1) max-flow computation.

[In fact they constructed a cut-tree (also known as Gomory-Hu tree) that gives all-pairs max-flow via (n-1) max-flow computation]

Max-flow :

Ford-Fulkerson $O(E|f_{max}|)$ [via augmenting path]

Edmonds-Karp $O(VE^2)$ [Augmenting path via BFS]

Push-relabel $O(V^2E)$

Chen et al. $O(E^{1+o(1)} \log U)$.

Maximum Flow and Minimum-Cost Flow in Almost-Linear Time

Li Chen* Georgia Tech lichen@gatech.edu
Rasmus Kyng† ETH Zurich kyng@inf.ethz.ch
Yang P. Liu‡ Stanford University yangpliu@stanford.edu
Richard Peng University of Waterloo § y5peng@uwaterloo.ca
Maximilian Probst Gutenberg† ETH Zurich maxprobst@ethz.ch
Sushant Sachdeva¶ University of Toronto sachdeva@cs.toronto.edu

April 26, 2022

71v2 [cs.DS] 22 Apr 2022

Abstract

We give an algorithm that computes exact maximum flows and minimum-cost flows on directed graphs with m edges and polynomially bounded integral demands, costs, and capacities in $m^{1+o(1)}$ time. Our algorithm builds the flow through a sequence of $m^{1+o(1)}$ approximate undirected minimum-ratio cycles, each of which is computed and processed in amortized $m^{o(1)}$ time using a new dynamic graph data structure.

Our framework extends to algorithms running in $m^{1+o(1)}$ time for computing flows that minimize general edge-separable convex functions to high accuracy. This gives almost-linear time algorithms for several problems including entropy-regularized optimal transport, matrix scaling, p-norm flows, and p-norm isotonic regression on arbitrary directed acyclic graphs.

Best paper FOCS'22

Quanta magazine | Physics | Mathematics | Biology | Computer Science | Topics | Archive

NETWORKS

Researchers Achieve 'Absurdly Fast' Algorithm for Network Flow

Computer scientists can now solve a decades-old problem in practically the time it takes to write it down.

⇒ $\tilde{O}(VE^{1+o(1)})$ time for (global) min cut.

Breaking the Cubic Barrier for All-Pairs Max-Flow: Gomory-Hu Tree in Nearly Quadratic Time

FOCS'22 paper

Amir Abboud* Robert Krauthgamer† Jason Li‡ Debmalya Panigrahi§
Thatchaphol Saranurak¶ Ohad Trabelsi‖

August 5, 2022

11.04958v3 [cs.DS] 3 Aug 2022

Abstract

In 1961, Gomory and Hu showed that the All-Pairs Max-Flow problem of computing the max-flow between all $\binom{n}{2}$ pairs of vertices in an undirected graph can be solved using only $n-1$ calls to any (single-pair) max-flow algorithm. Even assuming a linear-time max-flow algorithm, this yields a running time of $O(mn)$, which is $O(n^3)$ when $m = \Theta(n^2)$. While subsequent work has improved this bound for various special graph classes, no subcubic-time algorithm has been obtained in the last 60 years for general graphs. We break this longstanding barrier by giving an $\tilde{O}(n^2)$-time algorithm on general, weighted graphs. Combined with a popular complexity assumption, we establish a counter-intuitive separation: all-pairs max-flows are strictly *easier* to compute than all-pairs shortest-paths.

Our algorithm produces a cut-equivalent tree, known as the Gomory-Hu tree, from which the max-flow value for any pair can be retrieved in near-constant time. For unweighted graphs, we refine our techniques further to produce a Gomory-Hu tree in the time of a poly-logarithmic number of calls to any max-flow algorithm. This shows an equivalence between the all-pairs and single-pair max-flow problems, and is optimal up to poly-logarithmic factors. Using the recently announced $m^{1+o(1)}$-time max-flow algorithm (Chen *et al.*, March 2022), our Gomory-Hu tree algorithm for unweighted graphs also runs in $m^{1+o(1)}$-time.

Bangalore Theory Seminars

All-Pairs Minimum Cuts in Nearly Quadratic Time
Debmalya Panigrahy (Duke)

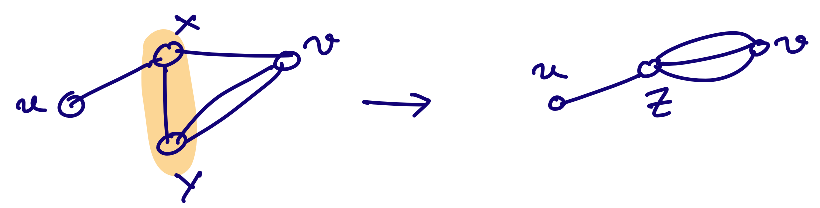
ICS: Thursday, 22 December 2022 10:00 AM-11:30 AM
YouTube Video Link

Abstract: In 1961, Gomory and Hu showed the surprising result that minimum cut

Option 2: Via Contractions.

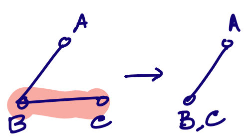
- A basic operation: contraction $e := (x, y)$;
- Replace (x, y) by a metavertex z.
  For $v \notin \{x, y\}$ replace $\{v, x\}$ by $\{v, z\}$
  replace $\{v, y\}$ by $\{v, z\}$.

No self loops!



$G \to G \setminus e.$

- Implementation of contraction. $O(n)$ time.



	A	B	C
A	0	1	0
B	1	0	1
C	0	1	0

→

	A	B	C
A	0	1	0
B	1	0	1
C	0	1	0

→

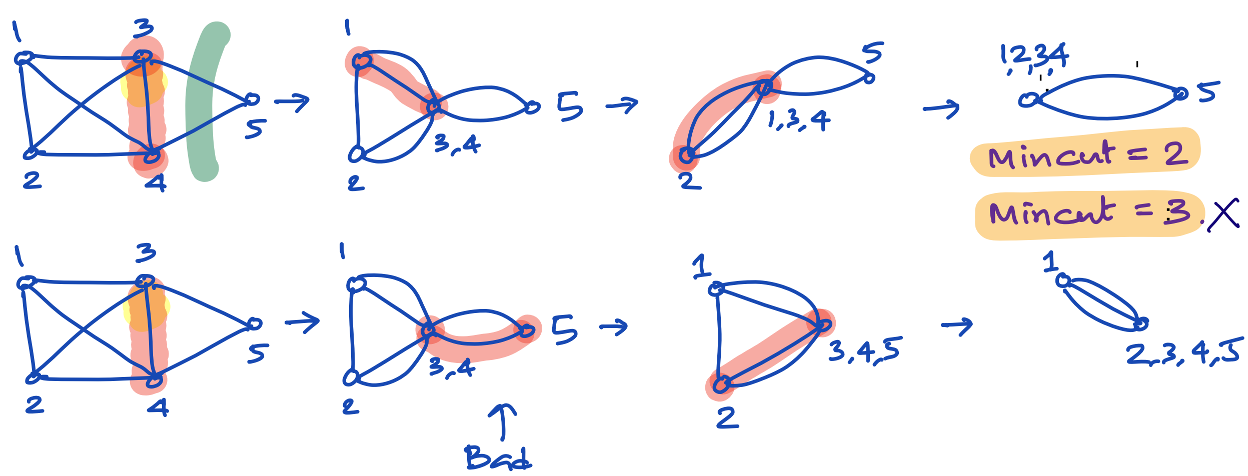
	A	B	C
A	0	1	0
B	1	0	0
C	0	0	0

→

	A	B,C
A	0	1
B,C	1	0

self loop reduction

merge neighbors



§ Randomized min-cut algorithm:

Karger [93] → Karger & Stein ['96] → Karger ['00]

$O(m \log^3 n)$

Monte Carlo Algorithm

Stoer & Wagner ['97]

Determ. $O(mn + n^2 \log n)$. ········→ $O(m^{1+o(1)})$.

Li

A New Approach to the Minimum Cut Problem

SODA'93
JACM'96

DAVID R. KARGER

Massachusetts Institute of Technology, Cambridge, Massachusetts

AND

CLIFFORD STEIN

Dartmouth College, Hanover, New Hampshire

Abstract. This paper presents a new approach to finding minimum cuts in undirected graphs. The fundamental principle is simple: the edges in a graph's minimum cut form an extremely small fraction of the graph's edges. Using this idea, we give a randomized, strongly polynomial algorithm that finds the minimum cut in an arbitrarily weighted undirected graph with high probability. The algorithm runs in $O(n^2 \log^3 n)$ time, a significant improvement over the previous $\tilde{O}(mn)$ time bounds based on maximum flows. It is simple and intuitive and uses no complex data structures. Our algorithm can be parallelized to run in $\mathcal{RNC}$ with n^2 processors; this gives the first proof that the minimum cut problem can be solved in $\mathcal{RNC}$. The algorithm does more than find a single minimum cut; it finds all of them.

With minor modifications, our algorithm solves two other problems of interest. Our algorithm finds all cuts with value within a multiplicative factor of α of the minimum cut's in expected $\tilde{O}(n^{2\alpha})$ time, or in $\mathcal{RNC}$ with $n^{2\alpha}$ processors. The problem of finding a minimum multiway cut of a graph into r pieces is solved in expected $\tilde{O}(n^{2(r-1)})$ time, or in $\mathcal{RNC}$ with $n^{2(r-1)}$ processors. The "trace" of the algorithm's execution on these two problems forms a new compact data structure for representing all small cuts and all multiway cuts in a graph. This data structure can be efficiently transformed into the more standard cactus representation for minimum cuts.

Minimum Cuts in Near-Linear Time

David R. Karger*

JACM'00

February 1, 2008

Abstract

We significantly improve known time bounds for solving the minimum cut problem on undirected graphs. We use a "semi-duality" between minimum cuts and maximum spanning tree packings combined with our previously developed random sampling techniques. We give a randomized algorithm that finds a minimum cut in an m-edge, n-vertex graph with high probability in $O(m \log^3 n)$ time. We also give a simpler randomized algorithm that finds *all* minimum cuts with high probability in $O(n^2 \log n)$ time. This variant has an optimal $\mathcal{RNC}$ parallelization. Both variants improve on the previous best time bound of $O(n^2 \log^3 n)$. Other applications of the tree-packing approach are new, nearly tight bounds on the number of *near minimum* cuts a graph may have and a new data structure for representing them in a space-efficient manner.

Deterministic Mincut in Almost-Linear Time

Jason Li*
Carnegie Mellon University

June 11, 2021

Abstract

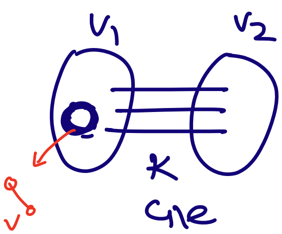
We present a deterministic (global) mincut algorithm for weighted, undirected graphs that runs in $m^{1+o(1)}$ time, answering an open question of Karger from the 1990s. To obtain our result, we de-randomize the construction of the *skeleton* graph in Karger's near-linear time mincut algorithm, which is its only randomized component. In particular, we partially de-randomize the well-known Benczur-Karger graph sparsification technique by random sampling, which we accomplish by the method of pessimistic estimators. Our main technical component is designing an efficient pessimistic estimator to capture the cuts of a graph, which involves harnessing the expander decomposition framework introduced in recent work by Goranci et al. (SODA 2021). As a side-effect, we obtain a structural representation of all approximate mincuts in a graph, which may have future applications.

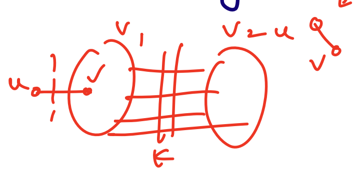
· Algorithm: ContractMC — $O(n^2)$ time.

1. $H := G$;
2. While H consists of > 2 vertices do:
   - choose $e \in E(H)$ uniformly at random
   - contract e, i.e. $H := H \backslash e$.
3. Let V_1, V_2 be vertex sets represented by the last two vertices in H. Return $[V_1, V_2]$.

• <u>Observation 0</u>: Let G be a multigraph. $\forall e \in E(G)$
$$c_{min}(G) \leq c_{min}(G \backslash e).$$

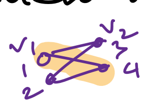
<u>Pf</u>: $[V_1, V_2]$ cut in $G\backslash e$ with k-edges is also a cut in G with k-edges.





Theorem: "ContractMC" outputs a mincut set with probability at least $2/(n(n-1))$.

<u>Pf</u>: Let k be the size of min-cut of G. We focus on one specific such min-cut $C = [V_1, V_2]$.



• <u>Observation 1</u>: $[V_1, V_2]$ is output of contractMC <u>iff</u> no edges between V_1 & V_2 is ever contracted.

Let e_i be the contracted edge in iteration i.
and $\mathcal{E}_i$ be the event that $e_i \notin C$.
Let $F_i := \bigcap_{j=1}^{i} \mathcal{E}_j$ be the event that no edges in C was contracted in the first i iterations.

We have to show $\mathbb{P}(F_{n-2}) \geq \frac{2}{n(n-1)}$.

First, Compute $\mathbb{P}(\mathcal{E}_1) = \mathbb{P}(F_1). = |C|/|E|$.

As $C_{min}(G) = K$, $\delta(G) \geq K$ where $\delta(G)$ is min degree of G.

$\therefore |E(G)| = \frac{\sum_{v \in V} deg(v)}{2} \geq \frac{nk}{2}$.

Out of $\frac{nk}{2}$ edges we dont want to select K edges in C. $\mathbb{P}[\bar{\mathcal{E}_1}] \leq K/(\frac{nK}{2})$.

As we select e_1 uniformly at random;
we have $\mathbb{P}[\mathcal{E}_1] = \mathbb{P}[F_1] \geq 1 - \frac{2K}{nK} = 1 - \frac{2}{n}$.

After first iteration, we are left with $(n-1)$ vertices. min cut value $\geq K$ (from obs. 0)

So, by as above

$$|E(G \setminus e_1)| \geq \frac{K(n-1)}{2}.$$

#edges might reduce due to self-loop reduction.

$$\mathbb{P}(\mathcal{E}_2 | F_1) \geq 1 - \frac{K.2}{K(n-1)} = 1 - \frac{2}{n-1}.$$

Similarly,

$$\mathbb{P}(\mathcal{E}_i | F_{i-1}) \geq 1 - \frac{K.2}{K(n-i+1)} = 1 - \frac{2}{(n-i+1)}.$$

Note: δ can increase, but that does not affect the analysis

$$\therefore\ \mathbb{P}(F_{n-2}) = \mathbb{P}(\mathcal{E}_{n-2} \cap F_{n-3}) \overset{\text{chain rule}}{=} \mathbb{P}(\mathcal{E}_{n-2} \mid F_{n-3}) \cdot \mathbb{P}(F_{n-3})$$

$$= \mathbb{P}(\mathcal{E}_{n-2} \mid F_{n-3}) \cdot \mathbb{P}(\mathcal{E}_{n-3} \mid F_{n-4}) \cdots \mathbb{P}(\mathcal{E}_2 \mid F_1) \cdot \mathbb{P}(F_1)$$

$$\geq \prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = \prod_{i=1}^{n-2} \left(\frac{n-i-1}{n-i+1}\right)$$

$$= \left(\frac{n-2}{n}\right) \cdot \left(\frac{n-3}{n-1}\right) \cdot \left(\frac{n-4}{n-2}\right) \cdots \left(\frac{2}{4}\right)\left(\frac{1}{3}\right) = \frac{2}{n(n-1)}.$$

- Power of repetitions.

Run contract MC independently for $n(n-1)\ln n$ times & output the min cut over all runs.

$\mathbb{P}$[failure after all run]

$= (\mathbb{P}\text{ of failure in one run})^{n(n-1)\ln n}$ (due to independence)

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

$\leq e^{-2\ln n}$ [$\because 1 - x \leq e^{-x}$ for $x > 0$]

$= \frac{1}{n^2} \to 0$ as $n \to \infty$. $\tilde{O}(n^4)$ time.

So, success prob. is boosted from $\frac{2}{n^2}$ to $(1 - \frac{1}{n^2})$. ∎

High probability: works w.p. $1 - O(1/n^c)$ for $c > 0$.

In general, let $1-p_1$ be $\mathbb{P}$ of failure on one run. We repeat $O(\frac{1}{p_1}\ln n)$ times to get $\frac{1}{n^{\Omega(1)}}$ prob of failure.

$(1-p_1)^{c\frac{1}{p_1}\ln n} \leq e^{-c\ln n} = \frac{1}{n^c}.$

As long as $\frac{1}{p_1} = \text{poly}(n)$, Runtime remains polynomial & we are happy.

- Does it work for the weighted graphs?

→ Yes! choose an edge for contraction w. probability proportional to the weight of the edge.

Q. Can we get faster algorithm?

Intuition: In the beginning, we make less error.

$$\mathbb{P}(F_{n-2}) = \mathbb{P}(\mathcal{E}_{n-2} \mid F_{n-3}) \cdot \mathbb{P}(\mathcal{E}_{n-3} \mid F_{n-4}) \cdots \mathbb{P}(\mathcal{E}_2 \mid F_1) \cdot \mathbb{P}(F_1)$$

$$\geq \prod_{i=1}^{n-2}\left(1 - \frac{2}{n-i+1}\right) = \left(\frac{n-2}{n}\right)\cdot\left(\frac{n-3}{n-1}\right)\cdot\left(\frac{n-4}{n-2}\right)\cdots\left(\frac{2}{4}\right)\left(\frac{1}{3}\right) = \frac{2}{n(n-1)}.$$

↑ larger ... ↑ smaller

Avoid!

- Lemma: Let C be a min-cut. Stop contraction when exactly t vertices are left. Then

$$\mathbb{P}[\text{no edge of } C \text{ is contracted}] \geq \frac{t(t-1)}{n(n-1)}.$$

Proof:

$$\prod_{i=1}^{n-t}\left(1 - \frac{2}{n-i+1}\right) = \left(\frac{n-2}{n}\right)\cdot\left(\frac{n-3}{n-1}\right)\cdot\left(\frac{n-4}{n-2}\right)\cdots\frac{n-(n-t)}{n-(n-t)+2}\cdot\frac{n-(n-t+1)}{n-(n-t+1)+2}$$

■

Approach 1: (Informal)

- (a) Contract till we are left with t vertices.
- (b) Then run usual contract MC on these t vertices, for ℓ parallel runs & return the best cut.

[Note: In the previous algo we had used $t = n$, $\ell = O(n^2 \ln n)$].

HW: Try to optimize ℓ & t [say, $\ell = O(n)$, $t = O(\sqrt{n})$] & then use parallel runs (if needed) to obtain $\tilde{O}(n^3)$ algorithm with high probability of success.

↑ (instead of $\tilde{O}(n^4)$ from Karger's algo)

- Karger - Stein faster algorithm :

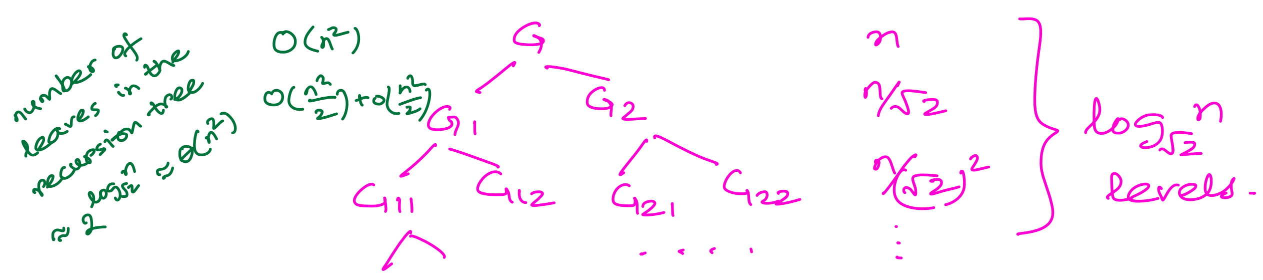
  Use above idea recursively.

Take $t = \frac{n}{\sqrt{2}} + 1$, then $\frac{t(t-1)}{n(n-1)} \geq \frac{\left(\frac{n}{\sqrt{2}}+1\right)\frac{n}{\sqrt{2}}}{n(n-1)} \geq \frac{1}{2} \cdot \frac{(n+\sqrt{2})}{(n-1)} \geq \frac{1}{2}$. . . . Fact 1.

- Algorithm: Fast-cut(G).

Input : Graph $G := (V, E)$ with n vertices & m edges.

1. If G has two metavertices corresponding to $(S, \bar{S})$,
   output $(S, \bar{S})$. → can even take any $O(1)$ & solve by brute-force.
2. For two times independently
   Run ContractMC until $\frac{n}{\sqrt{2}} + 1$ metavertices remain.
   Let G_1 and G_2 be two resultant multigraphs.
3. Recursively run Fast-cut on G_1 & G_2.
4. Return the best of Fast-cut (G_1) & Fast-Cut (G_2).



Theorem: Fast-cut has runtime $O(n^2 \log n)$.

$\#$contractions $= n - \frac{n}{\sqrt{2}} - 1$ (for step 2)

Each contraction takes $O(n)$ time.

$\Rightarrow$ steps due to contractMC takes $O(n^2)$ time.

There are two recursive calls.

$$T(n) = 2\left(T\left(\frac{n}{\sqrt{2}}\right) + O(n^2)\right) = O(n^2 \log n).$$

[$\rightarrow$ depth of search is $O(\log n)$, each step takes $O(n^2)$ time.]

Theorem: Fast-cut has success probability $\Omega(1/\log n)$.

Let $P(n)$ denote the success probability of Fast-cut(G).

$$P(n) = 1 - \mathbb{P}[\text{Both Fast-cut}(G_1) \text{ \& Fast-cut}(G_2) \text{ fail}]$$

$$= 1 - (1 - \mathbb{P}[\text{one branch is successful}])^2 \ldots$$

Now a branch is successful, when the contractMC for $n - n/\sqrt{2} - 1$ steps do not contract any edge in the min-cut [i.e. $\geq 1/2$ from Fact 1] and the recursive step is successful w.p. $P(n/\sqrt{2} + 1)$.

Hence, $P(n) = 1 - \left(1 - \frac{1}{2} P\left(\frac{n}{\sqrt{2}} + 1\right)\right)^2$

$$\approx 1 - \left(1 - \frac{1}{2} P\left(\frac{n}{\sqrt{2}}\right)\right)^2 \quad \text{[For simplicity]}$$

$$\Rightarrow P(n) = P\left(\frac{n}{\sqrt{2}}\right) - \frac{1}{4}\left[P\left(\frac{n}{\sqrt{2}}\right)\right]^2. \quad \ldots \circledast$$

Using induction & some algebraic manipulations one can show $P(n) \geq 4/\log n$. [HW]

- Intuition behind the recurrence:

$$p(n) = p(n/\sqrt{2}) - \frac{1}{4}[p(n/\sqrt{2})]^2$$

$$\Rightarrow p(n) - p(n/\sqrt{2}) = -\frac{1}{4}[p(n/\sqrt{2})]^2.$$

Replace, n by $\sqrt{2}^t$, i.e., $t = \log_{\sqrt{2}} n$.

then, $p(\sqrt{2}^t) - p(\sqrt{2}^{t-1}) = -\frac{1}{4}[p(\sqrt{2}^{t-1})]^2.$

Take $f(t) := p(\sqrt{2}^t) = p(n)$.

Then, $f(t) - f(t-1) = -\frac{1}{4}[f(t-1)]^2$

Intuitively, then

$$\frac{df}{dt} = -\frac{1}{4}f^2$$

$$\Rightarrow \frac{df}{f^2} = -\frac{1}{4}dt \quad \Rightarrow -\frac{1}{f} = -\frac{t}{4} \quad \text{[By integrating]}$$

$$\Rightarrow f(t) = \frac{4}{t} \Rightarrow p(n) = \frac{4}{t} = \frac{4}{\log_{\sqrt{2}} n} \approx \Omega\left(\frac{1}{\log n}\right).$$

The number of leaves in the recursion tree: $\Theta(n^2)$.

One can view this as another way of repeating contract MC $\Theta(n^2)$ times to amplify success probability.

However, runs are no more independent.
Different runs reuse same contraction step.

This saves a lot of runtime, and prob. of success drops little [constant to $\frac{1}{\log n}$].

Now we can repeat the algorithm $O(\log n)$ times, we get $O(n^2 \log^2 n)$ time with success probability:

$$1 - (1 - p(n))^{c\log n} = \Theta(1).$$

Repeat $O(\log^2 n)$ times & success prob $1 - \frac{1}{\log n}$.

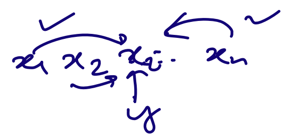
- Karger-Stein is one of those algorithms from THE BOOK, has many extensions, e.g. see the STOC'20 paper on K-cut by Gupta, Lee, Li.

Quicksort:

Algorithm RandomQS(S). $\{x_1, x_2, \ldots, x_n\} = S$.

Input: A set of n distinct numbers S.

Output: The elements of S sorted in increasing order.



1. If $|S| \leq 1$, return S. Else continue.
2. choose an element z (pivot) uniformly at random from S.
3. By comparing each element of S with z, determine the set L of elements smaller than y and the set R of elements greater than y.
4. Output: RandomQS(L), z, RandomQS(R).

Worst case: $\Theta(n^2)$. Consider $S = \{n, n-1, \ldots, 1\}$.

One can have $c(n) = c(n-1) + O(n)$. (#comparisons for array of n elements)

However if pivot splits S in a balanced way,

$$c(n) = c\left(\frac{n}{a}\right) + c\left(\frac{n(a-1)}{a}\right) + \Theta(n) \quad \text{for } a > 1.$$

we get $c(n) = O(n \lg n)$.

- Can we find good pivots often?

*without replacement.

- Option 1. Choose pivots uniformly at random. Here, expectation is over random choices of pivots. (randomized algo)

- Option 2. use <u>deterministic</u> algorithm & use first element as pivot; but take a random ordering of input. (rand ordering of input)

In this case, analysis of randomized Quicksort and probabilistic analysis of deterministic Quicksort under random inputs are essentially same. We focus on the rand. algo.

Secretary problem.

Thm 2.11: Expected number of comparisons made by Random Quicksort is $2n\ln n + O(n)$.

Pf: Let $y_1, y_2, \ldots, y_n$ be the same values as the input values $x_1, x_2, \ldots, x_n$ but sorted in increasing order.

X_{24}

$X = \{2, 5, 1, 3\}$ $Y = \{1, 2, 3, 5\}$.

For $i<j$, Let X_{ij} be a random variable that takes on value 1 if y_i and y_j are compared at some point in the algorithm & 0 otherwise.

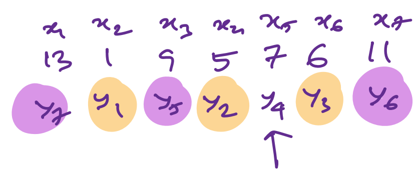
Hence, total no. of comparisons $X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$.

$\therefore \mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbb{E}[X_{ij}]$. (Lin of exp.)

X_{ij} is a Bernoulli RV.

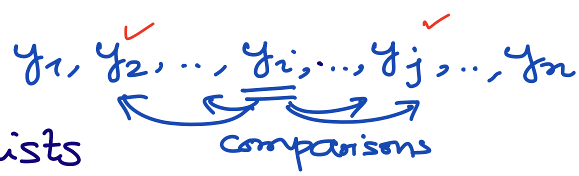
$\therefore \mathbb{E}[X_{ij}] = \mathbb{P}[y_i \,\&\, y_j \text{ are compared}] = p_{ij}$

Key idea: y_i and y_j are compared iff either y_i or y_j is the first pivot selected from the set $Y^{ij} = \{y_i, y_{i+1}, \dots, y_j\}$.



Pf of key idea: If y_i (or y_j) is the first pivot selected from this set, then y_i and y_j are in same sublist, and hence will be compared.

Otherwise they are separated into distinct sublists and so will not be compared. ∎



As pivots are chosen independently and uniformly at random. All members are equally likely to be selected. $p_{ij} = \frac{2}{j-i+1}$.

$$\therefore \mathbb{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{(j-i+1)}$$

$$= \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} \quad \text{[substituting } k = j-i+1\text{]}$$

$$= \sum_{k=2}^{n} \sum_{i=1}^{n+1-k} \frac{2}{k} \quad \text{[rearranging sums]}$$

$$= \sum_{k=2}^{n} (n+1-k) \frac{2}{k}$$

$$= \sum_{k=1}^{n} (n+1-k) \frac{2}{k} - 2n$$

$$= (n+1) \sum_{k=1}^{n} \frac{2}{k} - n.k.\frac{2}{k} - 2n$$

$$= (2n+2)\, H(n) - 4n. \approx O(n \log n). \blacksquare$$

i	k
1	2, 3, ..., n-2, n-1, n
2	2, 3, ..., n-2, n-1
3	2, 3, ..., n-2
:	:
(n-1)	2

$k + i \leq n+1$
$k \leq n-i+1.$
$i \leq n+1-k.$

Thm 2.12: Deals with probabilistic analysis.

⊙ Alternate analysis: Backwards Analysis of Quicksort

Very useful technique in many areas including computational geometry (randomized incremental construction).

Q. What is the expected number of comparisons in Quicksort.

Let $T(n)$ be the expected number of comparisons on an array of length n.

Now if we choose ith smallest element (y_i) then L & R has sizes $(i-1)$ & $(n-1)$, resp.

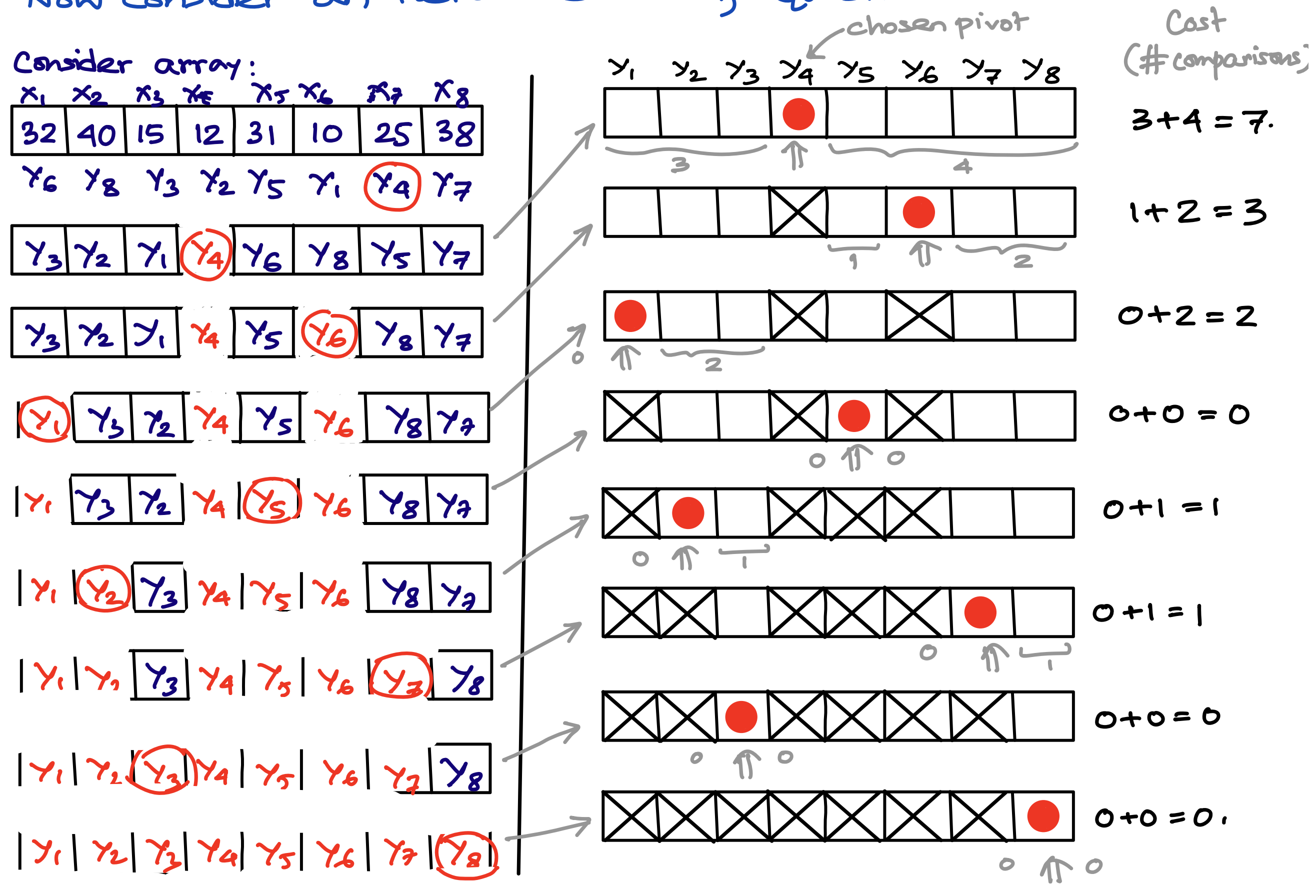
& we select y_i w.p. $1/n$.

So we obtain the following recurrence:

$$T(n) = \sum_{i=1}^{n} \mathbb{P}[y_i \text{ is chosen}] \cdot [(n-1) + T(i-1) + T(n-i)]$$

$$T(n) = \sum_{i=1}^{n} \frac{1}{n} [(n-1) + T(i-1) + T(n-i)], \quad T(0) = 0.$$

Now consider an iterative run of quicksort.



elements chosen in order: $y_4, y_6, y_1, y_5, y_2, y_7, y_3, y_8$.

25, 32, 10, 31, 12, 38, 15, 40.

This can be thought of an equivalent Dart Game.

1. Initially, there is a dart board of n consecutive, empty squares, arranged in a row.
2. For n iterations:

   Throw a dart at a uniformly random empty square, and

   pay cost = # consecutive empty squares to the left & right of the dart.

Note: After we throw the first dart, the empty segment to left & to right can be treated as two separate independent games.

There may no longer be a dart hitting the left segment every round, but conditioned on a dart hitting the left segment, the square it hits is still uniformly random.

HW: Show cost of this dart game is same as the # comparisons for quicksort.

Observation 1: The paid cost at a round is only specific to the present state. (does not depend on the history, i.e. how this state was reached).

Consider the following reversed (→ Backward) dart game:

1. Start with a full board of marked square.
2. For n iterations: unmark a random square each iteration & pay cost = # consecutive empty squares to the left & right of the unmarked square.

Observation 2: For any specific sequence of chosen squares in the original game, reversing the sequence in the reversed dart game arrives at the same cost per round (and thus total cost).

HW: verify cost of original game $[y_4 y_6 y_1 y_5 y_2 y_7 y_3 y_8]$

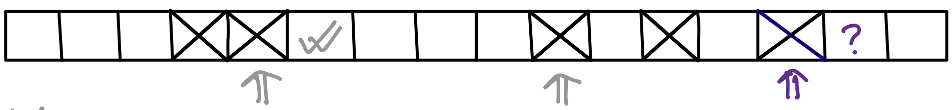
= Backward game with seq. $[y_8 y_3 y_7 y_2 y_5 y_1 y_6 y_4]$

Observation 3: Both sequence (permutation) occur with prob. $1/n!$ in their respective game, so they contribute the same amount to the expected costs of each game.

Advantage: Reversed game is much easier to analyze!

Think of cost being contribution from empty square.

→ In an iteration, when does a empty square contribute to the cost for this iteration?



'✓' empty square contributes when one of its two neighbor neighbor marked squares are unmarked [out of current i marked squares]

Hence, for each empty square, probability that it contributes to the cost on this round $\le 2/i$.

By linearity of expectations:

$$\mathbb{E}[\text{cost of } i\text{'th iteration}] \le \underbrace{(n-i)}_{\#\text{empty squares}} \cdot \frac{2}{i} = \frac{2n}{i} - 2.$$

Hence, total cost $\le \sum_{i=1}^{n} \left[\frac{2n}{i} - 2\right]$

$$= 2n \sum_{i=1}^{n} \frac{1}{i} - 2n = 2nH_n - 2n.$$

■

§ Types of randomized algorithms:

→ Monte Carlo algorithm (mostly correct)
probably correct ; guaranteed runtime.
e.g. Karger's mincut algorithm.

→ Las vegas algorithm
always correct ; expected runtime.
e.g. randomized quicksort.

§ Success amplification:

For MC: Perform k independent runs of MC algo (improve prob. of success)
Prob. of success amplifies from p to $1-(1-p)^k$.

For LV: Perform many independent runs of LV algo of $k\mathbb{E}[T]$ time, where $\mathbb{E}[T]$ is expected LV runtime. (improves runtime)

- LV → MC transformation:
Run LV algo for $k\mathbb{E}[T]$ times & halt, where T is the runtime of LV.
$\mathbb{P}[T > k\mathbb{E}[T]] < 1/k$. [from Markov's ineq – we'll see this later]
This MC version has runtime $k\mathbb{E}[T]$ and success probability $(1-1/k)$.

- Similarly, success prob. can be amplified by repetitions.

• One-sided vs two-sided error :

Randomized algorithms for decision problems can be :

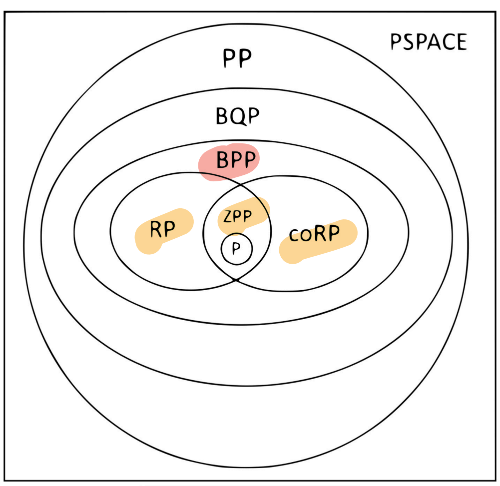
False-biased : always correct when it returns false.

True-biased : " " ... true.

Based on this we can define algorithms to have one-sided or two-sided errors.

HW : $p \to 1-(1-p)^k$ amplification worked for algorithms with one-sided error.
Find a strategy for amplifying success for algorithms with two-sided errors.

• Randomized complexity classes : → explain in terms of primality.



RP (Randomized Poly time)
→ consists of all languages L that have a poly-time randomized algo A s.t.

• If $x \notin L$, A always rejects x.
• If $x \in L$, A accepts w.p. $\geq 1/2$.

Co-RP (Complement of RP)

• If $x \in L$, A always accepts x.
• If $x \notin L$, A rejects w.p. $\geq 1/2$.

So, RP/Co-RP corresponds to MC algos w. one-sided error.

[If a problem admits LV algo then we can convert it to MC algo w. one-sided error].

Note: $1/2$ is just a placeholder. we can make it any constant > 0. (think about amplification of prob. of success)

ZPP (Zero-error probabilistic polynomial time)

- If $x \in L$, $\mathcal{A}$ always accepts x.
- If $x \notin L$, $\mathcal{A}$ always rejects x.

} runs in exp. poly-time.

- Corresponds to Las Vegas algorithm.

[Here, as we need zero-error, we can't use MC algorithms]

Theorem: ZPP = RP $\cap$ Co-RP. ← HW

[If $x \in$ RP $\cap$ Co-RP, then $x \in$ ZPP.
Other direction is more tricky!]

BPP (Bounded-error probabilistic polynomial time)

- If $x \in L$, $\mathcal{A}$ accepts x w.p. $\geq 3/4$.
- If $x \notin L$, $\mathcal{A}$ accepts x w.p. $\leq 1/4$.

Both-sided error.

- Known: RP $\subseteq$ NP, CoRP $\subseteq$ CoNP.
  Conjecture: BPP $\subseteq$ NP? BPP = P?

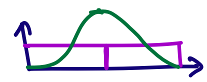
- Polynomial Identity Testing (PIT) is in BPP (also co-RP), but not known to be in P.

$(x+y)(x-y) \overset{?}{=} (x^2-y^2)$.

Best resource for learning about complexity classes:

→ Computational Complexity by Arora-Barak [Ch 7: Randomized Computation].

3 Moments and Deviation.



• Variance and moments of a random variable.

- k'th moment of a RV X is $\mathbb{E}[X^k]$. *like kth derivative for functions*
- variance of X: $\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}(X))^2] = \mathbb{E}[X^2] - (\mathbb{E}(X))^2$.
- covariance of RVs X & Y:

  $\mathrm{Cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}X)(Y - \mathbb{E}Y)]$.

• Thm 3.2: $\mathrm{Var}[X+Y] = \mathrm{Var}(X) + \mathrm{Var}(Y) + 2\,\mathrm{Cov}(X,Y)$

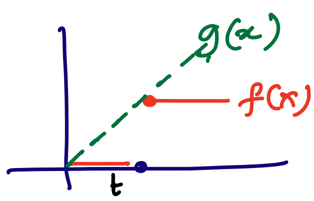
• If X, Y are indep then $\mathrm{Cov}(X,Y) = 0$ and Thm 3.2 is like linearity of expectation.

• Thm 3.3: $\mathbb{E}[X \cdot Y] = \mathbb{E}[X] \cdot \mathbb{E}[Y]$ for indep X, Y.

Theorem 3.1 [Markov's inequality]

For a nonnegative random variable X, $\forall\, t > 0$

$$\boxed{\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.} \quad \text{or} \quad \Pr[X \geq t\,\mathbb{E}X] \leq \frac{1}{t}.$$



Define $f(x) = \begin{cases} 0 & \text{for } x < t \\ 1 & \text{for } x \geq t \end{cases}$,

Define $g(x) = x/t$.

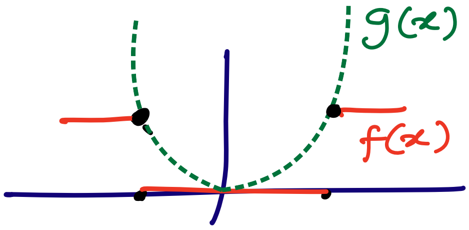
Fact 1. $g(x) \geq f(x)$.

Fact 2. $\mathbb{E}[f(X)] = 0 \cdot \Pr[X < t] + 1 \cdot \Pr[X \geq t] = \Pr[X \geq t]$.

$$\begin{aligned}
\Pr[X \geq t] &= \mathbb{E}[f(X)] && \text{[Fact 2]} \\
&\leq \mathbb{E}[g(X)] && \text{[Fact 1]} \\
&= \mathbb{E}[X/t] \\
&= \frac{1}{t}\mathbb{E}[X]. && \text{[scaling]}
\end{aligned}$$

Theorem 3.2 [Chebyshev's inequality]

For any $a>0$, $\mathbb{P}[|X-\mathbb{E}X|>a] \leq \frac{Var[X]}{a^2}$.



Wlog assume $\mathbb{E}X=0$, $Var X = 1$. (By scaling)

So, $\mathbb{E}[X^2]=1$.

Define $f(x) = 0$ for $|x|<t$
$\quad\quad\quad\;\; = 1$ for $|x| \geq t$.

$g(x) = \frac{x^2}{t^2}$.

$$\Pr[|X| \geq t] = \mathbb{E}[f(X)]$$
$$\leq \mathbb{E}[g(X)] \quad [\text{As, } g(x) \geq f(x)]$$
$$= \mathbb{E}\left[\frac{X^2}{t^2}\right] = \frac{1}{t^2}\mathbb{E}[X^2] = \frac{1}{t^2} 1.$$

· Application:

$x_i = \begin{cases} 1 & \text{if } i\text{'th coin flip is head} \\ 0 & \text{else.} \end{cases}$ | $X = \sum_1^n x_i$ denote # heads in n coin flips.

$\therefore \mathbb{E}[X] = np = n/2$, $Var[X] = \sum Var(x_i) = n \cdot \frac{1}{4}$.

Markov: $\mathbb{P}(X \geq 3n/4) \leq \frac{\mathbb{E}X}{3n/4} = \frac{n/2}{3n/4} = \frac{2}{3}$.

↑ simple to use. sometimes tight.

Chebyshev: $\mathbb{P}(X \geq 3n/4) \leq \mathbb{P}(|X - \mathbb{E}X| \geq n/4)$

$$\leq \frac{Var(X)}{(n/4)^2} = \frac{n/4}{(n/4)^2} = \frac{4}{n}.$$

↑ much better bound when var. is low.