Portland State University

W'21 CS 584/684

Algorithm Design & Analysis

Fang Song

Lecture 19

- Hamiltonian Cycle
- Approximation algorithms
- Randomized algorithms

Establishing NP-Completeness

Once we establish first "natural" NP-complete problem, others fall like dominoes …

Recipe to establish NP-Completeness of problem Y

1. Show that $Y \in \mathbf{NP}$
2. Choose an $\mathrm{NP-complete}$ problem X
3. Prove that $X \leq_{P,Karp} Y$

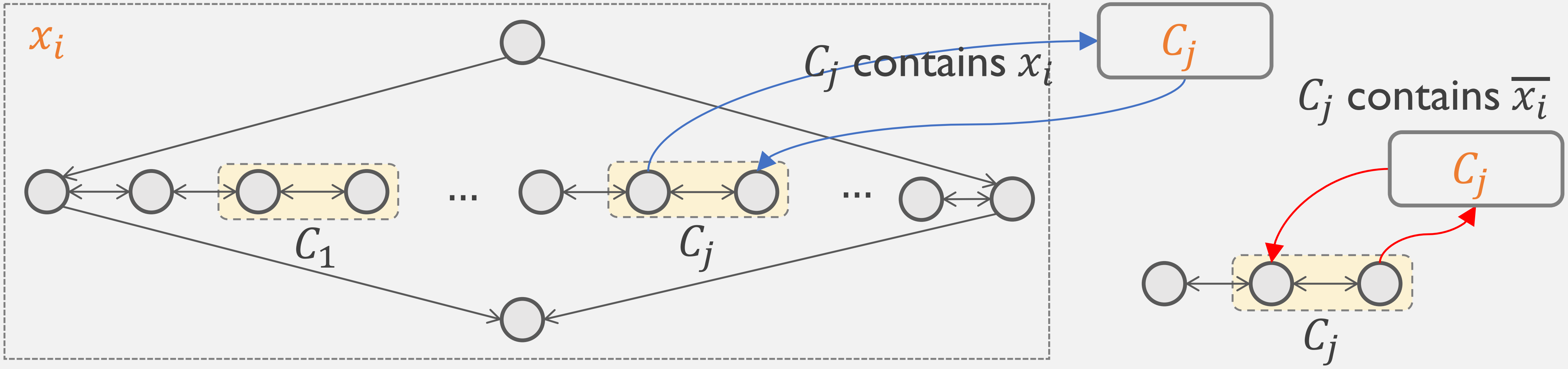
Justification. If X is an NP-complete problem, and Y is a problem in $\mathbf{NP}$ with the property that $X \leq_{P,Karp} Y$ then Y is NP-complete (by transitivity)

(DIR−)HAM−CYCLE is NP-Complete

(DIR−)HAM−CYCLE. Given a directed graph $G = (V, E)$, does there exist a directed cycle Γ that visits every node exactly once?

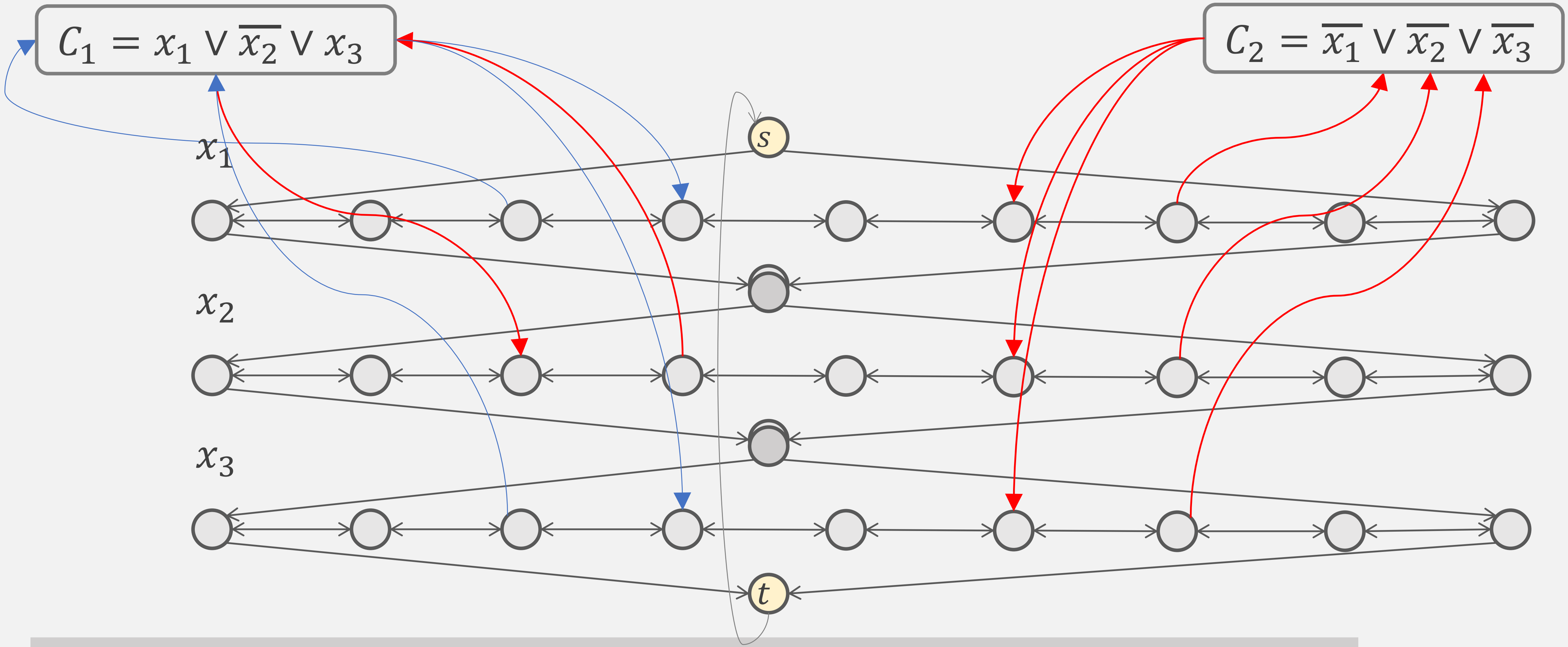
Theorem. $\text{3−SAT} \leq_P \text{(DIR−)HAM−CYCLE}$

Pf. Given 3−SAT instance Φ in CNF: n variables x_i and k clauses C_j



Intuition: traverse row i from left to right $\Leftrightarrow$ set variable x_i = true

$3\text{–SAT} \leq_P (\text{DIR–})\text{HAM–CYCLE}$



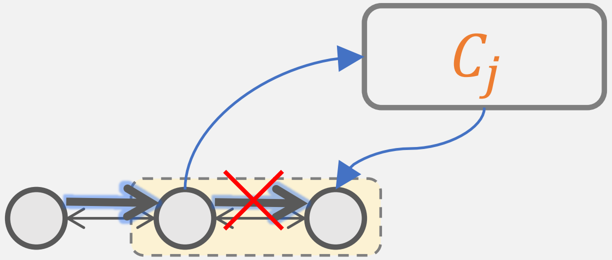
Claim. Φ is satisfiable iff. G has a Hamiltonian cycle

$3\text{−SAT} \le_P (\text{DIR−})\text{HAM−CYCLE}$

Claim. Φ is satisfiable iff. G has a Hamiltonian cycle

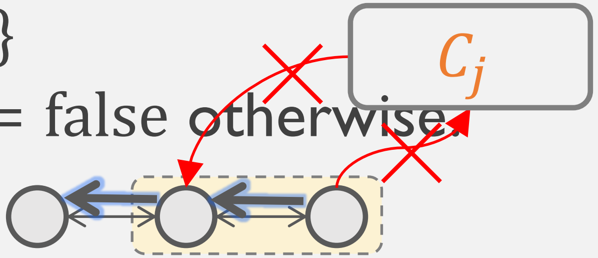
($\Rightarrow$) Suppose Φ has a satisfying assign. $\boldsymbol{x}^*$. Define an H-Cycle in G:

- if $x_i^* = \text{true}$, traverse row x_i from left to right
- if $x_i^* = \text{false}$, traverse row x_i from right to left
- For each clause C_j pick (only) one row i and take a detour



($\Leftarrow$) Suppose G has a H-Cycle Γ. Define a satisfying assign. in Φ:

- In Γ, replace edges going/leaving C_j with the edge of the corresponding two nodes in some row. This gives a new cycle Γ' in $G - \{C_1, C_2, \ldots, C_k\}$
- In Γ', set $x_i = \text{true}$ if Γ' traverses row i left-to-right; set $x_i = \text{false}$ otherwise.

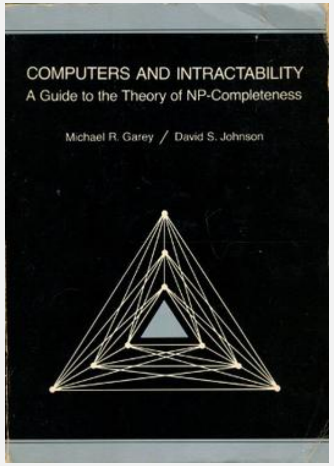


Hard computational problems cont'd

- Aerospace engineering: optimal mesh partitioning for finite elements.
- Chemical engineering: heat exchanger network synthesis
- Civil engineering: equilibrium of urban traffic flow
- Electrical engineering: VLSI layout.
- Mechanical engineering: structure of turbulence in sheared flows
- Biology: protein folding
- Physics: partition function of 3-D Ising model in statistical mechanics.
- Economics: computation of arbitrage in financial markets with friction
- Financial engineering: find minimum risk portfolio of given return
- Politics: Shapley-Shubik voting power
- Pop culture: Sudoku (http://www-imai.is.s.u-tokyo.ac.jp/~yato/data2/SIGAL87-2.pdf)

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Want to learn more?

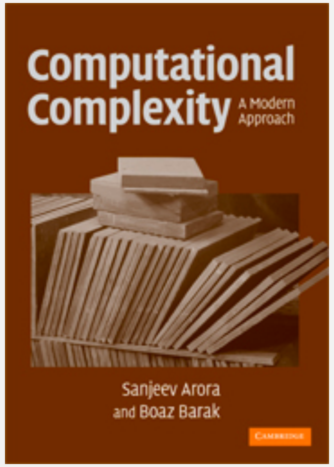


Computers and Intractability: A Guide to the Theory of NP-Completeness.
Michael Garey and David S. Johnson

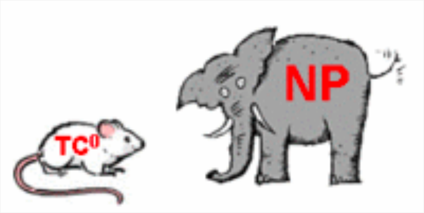
Most Cited Computer Science Citations

This list is generated from documents in the CiteSeer^x database as of March 19, 2015. This list is automaticall
mode and citation counts may differ from those currently in the CiteSeer^x database, since the database is con
All Years | 1990 | 1991 | 1992 | 1993 | 1994 | 1995 | 1996 | 1997 | 1998 | 1999 | 2000 | 2001 | 2002 | 2003 | 20
| 2015

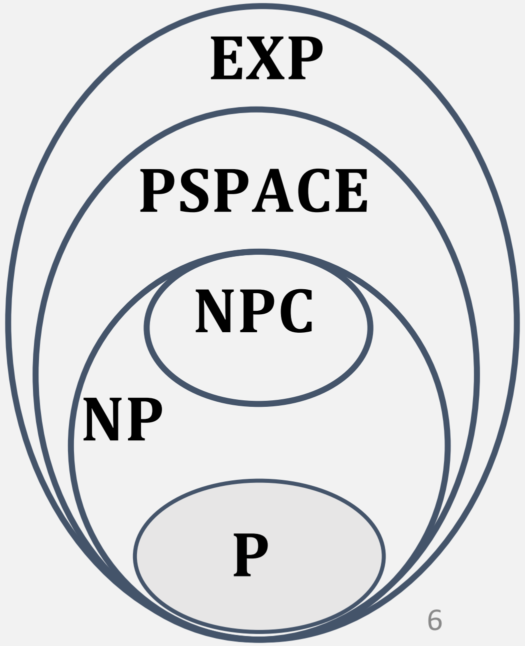
1. M R Garey, D S Johnson
Computers and Intractability: A Guide to the Theory of NPCompleteness" W.H. Feeman and 1979
11468



Computational Complexity: A Modern Approach
Sanjeev Arora & Boaz Barak



Complexity Zoo
There are now 544 classes and counting!



Coping with NP-Completeness

“I can’t find an efficient algorithm, I guess I’m just too dumb.”

“I can’t find an efficient algorithm, but neither can all these famous people.”

https://i.stack.imgur.com/EkpIV.jpg

- Better (constructive) answers:
  sacrifice one of three desired features

  1. ~~Solve arbitrary instances~~
  2. ~~Solve problems in poly-time~~
  3. ~~Solve problems to optimality~~

- Techniques
  - Identifying structured special cases
  - Local search heuristics (e.g., gradient descent)
  - Approximation algorithms

Finding near-optimal vertex cover

Input. Graph $G = (V, E)$

- Vertex cover $S \subseteq V$: subset of vertices that touches all edges

Goal. Find a vertex cover S of minimum size

- First attempt: greedily pick the vertex that covers most edges

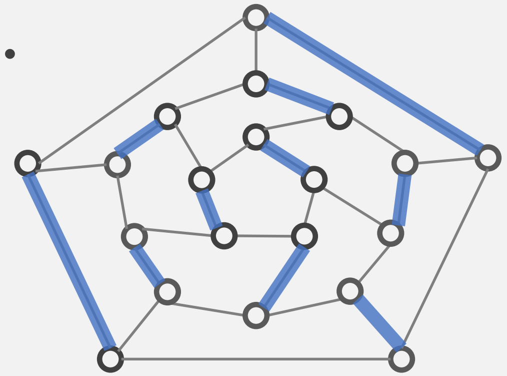
> APP-VC: on input $G = (V, E)$
> For $v \in V$ (in descending order of degrees)
> Add v in S
> Delete v and its neighbors from G

- Claim. Suppose the minimum vertex cover has size OPT. Then the output of APP-VC has size at most $O(\log n \cdot \mathrm{OPT})$
- Pf. Exercise

2-approximation vertex cover

Recall: $M \subseteq E$ is a matching in $G = (V, E)$ if each node appears in at most one edge in M.

Observation: For any matching M and any vertex cover S, $|M| \leq |S|$. In particular, $|M| \leq \mathrm{OPT}$ (size of min vertex cover).



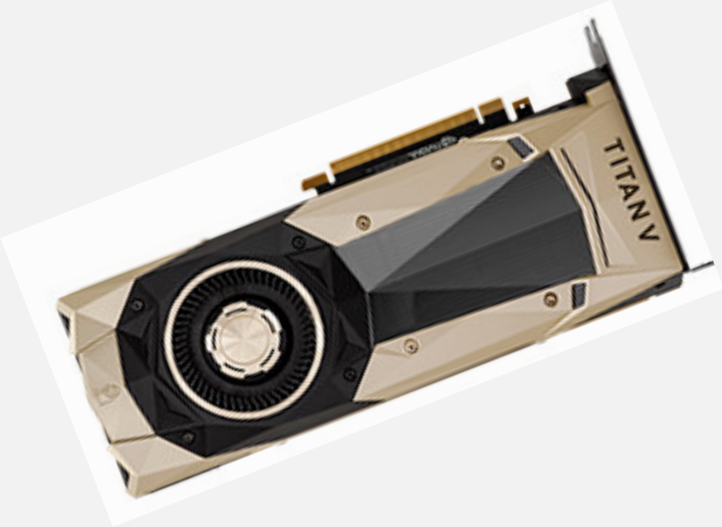
- 2nd attempt: find a MAX matching

> 2-APP-VC: on input $G = (V, E)$
> Find a maximal matching $M \subseteq E$
> Return $S = \{\text{all end points of edges in } M\}$

- Claim. The output of 2-APP-VC has size at most $2 \cdot \mathrm{OPT}$
- Pf. $|S| = 2|M| \leq 2 \cdot \mathrm{OPT}$. Why does S have to be a vertex cover?
  - Exercise. Is this tight, i.e., 2-APP-VC's output $= 2 \cdot \mathrm{OPT}$ on some graph?

Scarce computational resources, which to invest on?

www.flickr.com



www.nvidia.com

www.computerhope.com

How about … coins?

Theorem. Randomness is useful

- **Randomization.** Allow fair coin flip in unit time

Power of randomness: primality testing

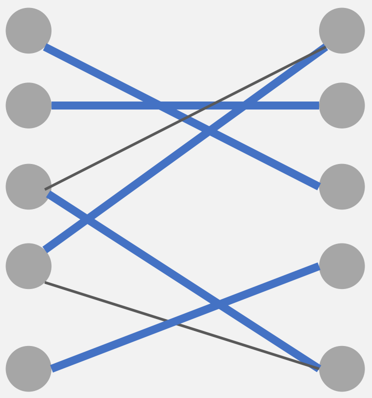
Is integer n Prime?

20,988,936,657,440,586,486,151,264,256,610,222,593,863,921

- Naive method: $O(n)$
- Randomized algorithm: Miller-Rabin 1977 $O(\log^4 n)$
- Deterministic algorithm: AKS 2002 $O(\log^{12} n)$

Miller-Rabin is still the way to go in practice!

Power of randomness: perfect matching



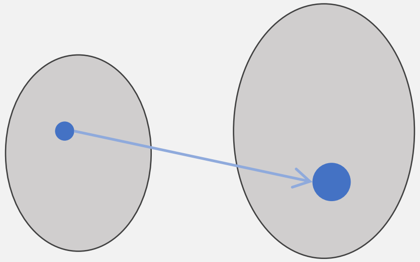
m: # edges
n: # nodes

- Deterministic algorithm: $O(nm)$
- Randomized algorithm: $O(\log^c nm)$

Exponentially faster!

Power of randomness beyond algorithm design

Probabilistic constructions



Nice error-correction codes exist:
random codes

Cryptography

Probabilistic Encryption*

Shafi Goldwasser and Silvio Micali

Probability 101

- (Discrete) Sample space $\Omega = \{\omega\}$
  - set of all possible outcomes of a random experiment
  - Event $E \subseteq \Omega$: a subset of the sample space
- Axioms of probability: a probability distribution is a mapping from events to real numbers $\Pr(\cdot): \mathcal{P}(\Omega) \to [0,1]$, satisfying
  - Probability of an event $\Pr(E) \geq 0$ for any event E
  - $\Pr(\Omega) = 1$
  - $\Pr(E \cup F) = \Pr(E) + \Pr(F)$ if $E \cap F = \emptyset$ (mutually exclusive)
- Ex. Roll a fair dice
  - $\Omega = \{1,2,3,4,5,6\}, \Pr(\omega) = \frac{1}{6}, \omega = 1, \dots, 6.$
  - $E = \{1,3,5\}$ dice being odd, & $\Pr(E) = 1/2$

N.B. $\bar{E} \coloneqq \Omega \backslash E$ complement event
$\Pr(\bar{E}) = 1 - \Pr(E)$

Probability 101 cont'd

- **Conditional probability:** $\Pr(B|A) \coloneqq \frac{\Pr(A \cap B)}{\Pr(A)}$, assuming $\Pr(A) > 0$.

Bayes' theorem

Let E, F be two events and $\Pr(F) > 0$.

Then $\Pr(E|F) = \Pr(F|E) \cdot \frac{\Pr(E)}{\Pr(F)}$.

- **Independence:** Events A, B are independent iff. $\Pr(B|A) = \Pr(B)$.

i.e. $\Pr(A \cap B) = \Pr(A) \cdot \Pr(B)$

Probability 101 cont'd

- Random variable $X: \Omega \to \mathbb{N}$
  - Assign each outcome a number
  - "$X = x$" is the event $E := \{\omega \in \Omega: X(\omega) = x\}$
  - Independent random variables:

X, Y are indep. iff. for all possible x and y, events $X = x$ and $Y = y$ are indep.

- Expectation: a weighed average
  - $\mathbb{E}[X] = \sum_{z \in Z} \Pr(X = z) \cdot z$
  - Linearity: $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ (independence NOT needed)
- Ex. $\Omega =$ roll 4 dices independently
  - Let X be the sum of 4 rolls; X_i be value of ith roll, $i = 1, \ldots, 4$
  - $\mathbb{E}[X] = \mathbb{E}[X_1 + \cdots + X_4] = 4 \cdot \mathbb{E}[X_1] = 4 \times 3.5 = 14$

Recall: quick sort

- Main Idea
  - Divide array into two halves.
  - Recursively sort each half.
  - Merge two halves to make sorted whole.

with condition: $L \leq pivot \leq R$

trivially

- Analysis
  - Correctness
  - Running time*

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

Cost in divide, not merge

* best-case partition

- Can you think of a worst-case scenario?

Randomized quicksort

- Pick the pivot randomly

```
Rand-QuickSort(A):
  if (array A has zero or one element)
     Return
  Pick pivot p ∈ A uniformly at random
  (L, M, R) ← PARTITION − 3 − WAY(A, p)      → O(n)
  Rand-QuickSort(L)                           → T(i)
  Rand-QuickSort(R)                           → T(n − i − 1)
```

Theorem. The expected number of compares to quicksort an array of n distinct elements is $O(n \log n)$.

Randomized quicksort: analysis

Theorem. The expected number of compares to quicksort an array of n distinct elements is $O(n \log n)$.

Assume $A = \{z_1, z_2, \dots, z_n\}, z_1 < z_2 < \cdots < z_n$

Observation: any pair z_i & z_j $(i < j)$ is compared at most once

- How many comparisons? $X :=$ total number of comparisons
  - Indicator variable: $X_{ij} := \begin{cases} 1, \text{if } z_i \text{ is compared to } z_j \\ 0, \text{otherwise} \end{cases}$

$$\Rightarrow \mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right]$$
$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathbb{E}[X_{ij}] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr[X_{ij} = 1]$$

Linearity

Randomized quicksort: analysis cont'd

Theorem. The expected number of compares to quicksort an array of n distinct elements is $O(n \log n)$.

$$\boxed{\mathbb{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr[X_{ij} = 1]} \qquad X_{ij} := \begin{cases} 1, \text{if } z_i \text{ is compared to } z_j \\ 0, \text{otherwise} \end{cases}$$

- When two items are compared?

$$2 \quad 1 \quad 3 \quad \underbrace{4 \quad 6 \quad 5 \qquad \boxed{7} \qquad 9 \quad 8}_{\text{No comparison across these two groups}} \quad 10$$

- Observation: z_i & z_j compared iff. z_i or z_j was the first chosen as a pivot from $Z_{ij} = \{z_i, z_{i+1}, \dots, z_j\}$

Randomized quicksort: analysis cont'd

- Observation: z_i & z_j compared iff. z_i or z_j was the first chosen as a pivot from $Z_{ij} = \{z_i, z_{i+1}, \dots, z_j\}$

$$
\begin{aligned}
&\Pr[X_{ij} = 1] \\
&= \Pr[z_i \,\&\, z_j \text{ compared}] = \Pr[z_i \text{ or } z_j \text{ is 1st pivot chosen from } Z_{ij}] \\
&= \Pr[z_i \text{ is 1st pivot from } Z_{ij}] + \Pr[z_j \text{ is 1st pivot from } Z_{ij}] \\
&= \frac{1}{j-i+1} + \frac{1}{j-i+1} = \frac{2}{j-i+1}
\end{aligned}
$$

$$\mathbb{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \leq 2 \cdot \sum_{i=1}^{n-1} \boxed{\sum_{k=1}^{n} \frac{1}{k}} = O(n \cdot \log n)$$

Harmonic series

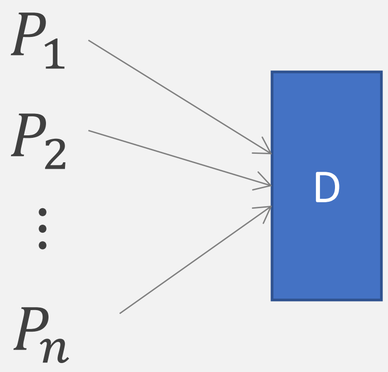
Contention resolution in a distributed system

Given: processes $P_1, \dots, P_n$,

- each process competes for access to a shared database.
- If ≥ 2 processes access the database simultaneously, all processes are locked out.

Goal: a protocol so all processes get through on a regular basis

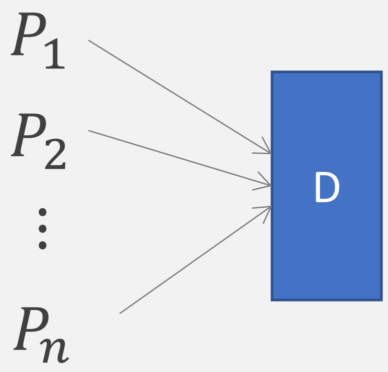
- Restriction: Processes can't communicate.



Contention resolution: randomized protocol

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

Theorem. All processes will succeed in accessing the database *at least once* within $O(n \ln n)$ rounds except with probability $\leq \frac{1}{n}$.



Randomized contention resolution: analysis 1

Def. $S[i,t]$ = event that process i succeeds in accessing the database in round t.

- Claim1. $\frac{1}{e \cdot n} \leq \Pr(S[i,t]) \leq \frac{1}{2n}$
- Pf. $\Pr(S[i,t]) = p(1-p)^{n-1}$

[Geometric distribution: independent Bernoulli trials]

Process i requests access

None of remaining request access

$\Rightarrow \Pr(S[i,t]) = \frac{1}{n}(1-1/n)^{n-1} \in [\frac{1}{en}, \frac{1}{2n}]$ $[p = 1/n]$

- $(1 - 1/n)^n$ converges monotonically from $1/4$ up to $1/e$.
- $(1 - 1/n)^{n-1}$ converges monotonically from $1/2$ down to $1/e$.

Randomized contention resolution: analysis 2

- Claim2. The probability that process i fails to access the database in $e \cdot n$ 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.

$$\Pr(F[i,t]) = \Pr\left(\overline{S[i,1]}\right) \cdot \ldots \cdot \Pr\left(\overline{S[i,t]}\right) \leq \left(1 - \frac{1}{en}\right)^{t}$$

[Independence & Claim 1]

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

Randomized contention resolution: analysis 3

Theorem. All processes will succeed in accessing the database *at least once* within $2en \ln n$ rounds except with probability $\leq \frac{1}{n}$.

- Pf. Let $F[t]$ = event that some process fails to access database in rounds 1 through t.

Union Bound

Let E, F be two events. Then $\Pr(E \cup F) \leq \Pr(E) + \Pr(F)$.

$$\Pr(F[t]) = \Pr(\cup_{i=1}^{n} F[i,t]) \leq \sum_{i=1}^{n} \Pr(F[i,t]) \leq n \cdot \Pr(F[1,t])$$

  - Choose $t = en \cdot 2\ln n$: $\Pr(F[t]) \leq n \cdot n^{-2} = 1/n$

Integer linear programming (ILP)

Input. Graph $G = (V, E)$

- Vertex cover $S \subseteq V$: subset of vertices that touches all edges

Goal. Find a vertex cover S of minimum size

- Formulating vertex cover as an integral linear program

For each $i \in V$, introduce $x_i \in \{0,1\}$
Min $\sum_{i=1}^{n} x_i$
Subject to:
$$x_i + x_j \geq 1 \quad \text{for each } (i,j) \in E$$

[i.e., Pick i in vertex cover iff. $x_i = 1$]

☹ We don't know (expect) a poly-time algorithm (ILP)

- Without integrality (LP), we do know poly-time algorithms

Putting aside the integral constraint

(ILP Π) Min $\sum_{i=1}^{n} x_i$
Subject to:

$$x_i + x_j \geq 1, \qquad \forall (i,j) \in E$$
$$x_i \in \{0,1\}, \qquad \forall i \in V$$

(LP Σ) Min $\sum_{i=1}^{n} x_i$
Subject to:

$$x_i + x_j \geq 1, \qquad \forall (i,j) \in E$$
$$0 \leq x_i \leq 1, \qquad \forall i \in V$$

?

Let x^* be an optimal soln. for LP Σ
& optimal value $\mathrm{OPT} = \sum_i x_i^*$

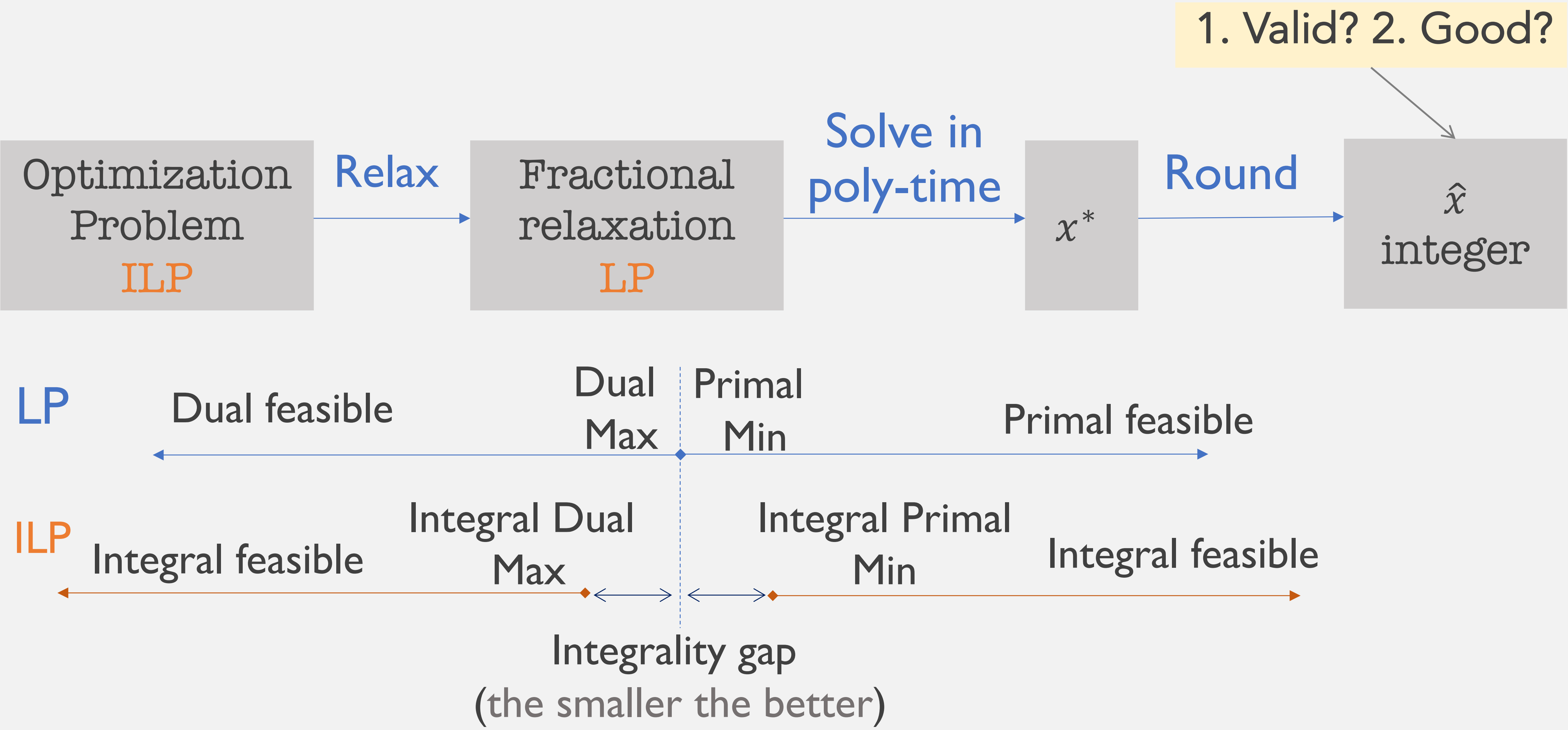
$$x_i := \lfloor x_i^* \rceil = \begin{cases} 1, & \text{if } x_i^* \geq \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}$$

- (Threshold) Rounding:

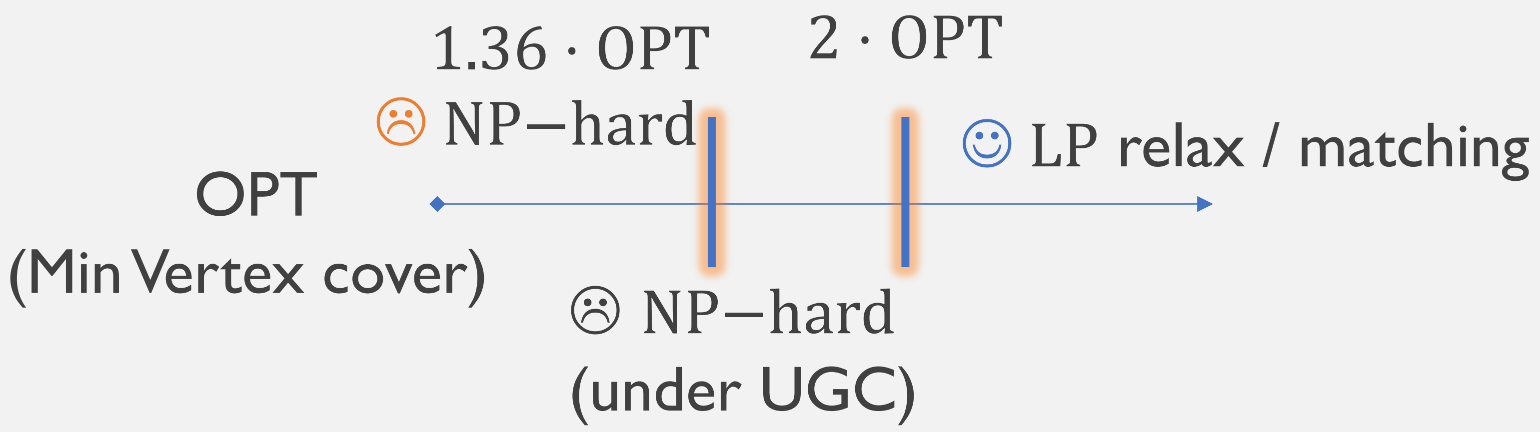
i. $\{x_i\}$ is a feasible integral solution: $\forall (i,j) \in E, x_i^* \geq \frac{1}{2}$ or $x_j^* \geq \frac{1}{2}$ or both

ii. $\sum_i x_i \leq \sum_i 2 \cdot x_i^* = 2 \cdot \mathrm{OPT} \leq 2 \cdot \mathrm{OPT}_{\mathrm{Int}}$ [optimal value of ILP Π, i.e. size of min vertex cover]

LP relaxation



Hardness of approximation



Theorem. It is NP-Hard to approximate Vertex Cover to with any factor below 1.36067. [i.e., otherwise, you can solve 3-SAT in poly-time]

Theorem'. It is NP-Hard to approximate Vertex Cover to with any factor below 2, assuming the unique games conjecture (UGC).

Want to read more?

https://cs.nyu.edu/~khot/papers/UGCSurvey.pdf

https://cs.stanford.edu/people/trevisan/pubs/inapprox.pdf

Approximating set cover

Input. Set U of n elements, $S_1, \dots, S_m$ of subsets of U

Goal. Find $I \subseteq \{1, \dots, m\}$ of minimum size such that $\bigcup_{i \in I} S_i = U$

(ILP Π for Set cover)

For each $i \in \{1, \dots, m\}$, introduce $x_i \in \{0,1\}$

Min $\sum_{i=1}^{m} x_i$

Subject to:

$$\sum_{i: u \in S_i} x_i \geq 1, \qquad \forall u \in U$$

LP relaxation for set cover

(Set cover ILP Π)
Min $\sum_{i=1}^{m} x_i$
Subject to:

$$\sum_{i: u \in S_i} x_i \geq 1, \qquad \forall u \in U$$
$$x_i \in \{0,1\}, \qquad \forall i \in \{1, \dots, m\}$$

(Set cover Σ)
Min $\sum_{i=1}^{m} x_i$
Subject to:

$$\sum_{i: u \in S_i} x_i \geq 1, \qquad \forall u \in U$$
$$0 \leq x_i \leq 1, \forall i \in \{1, \dots, m\}$$

Let x^* be an optimal soln. for LP Σ
& optimal value $\mathrm{OPT} = \sum_i x_i^*$

? $x_i := \lfloor x_i^* \rceil$

- Threshold rounding: does it cover all elements?
  - Ex. $u \in S_1, \dots, S_{100}; x_1^*, \dots x_{100}^* = \frac{1}{100} \Rightarrow x_1 = \cdots = x_{100} = 0$. u is missed!
- Randomized rounding!

LP relaxation for set cover

(Set cover ILP Π) Min $\sum_{i=1}^{m} x_i$
Subject to:
$$\sum_{i:u\in S_i} x_i \geq 1, \ \forall u \in U$$
$$x_i \in \{0,1\}, \qquad \forall i \in \{1,\ldots,m\}$$

(Set cover LP Σ) Min $\sum_{i=1}^{m} x_i$
Subject to:
$$\sum_{i:u\in S_i} x_i \geq 1, \ \forall u \in U$$
$$0 \leq x_i \leq 1, \forall i \in \{1,\ldots,m\}$$

Let x^* be an optimal soln. for LP Σ & optimal value $\mathrm{OPT} = \sum_i x_i^*$

☹ $x_i := \lfloor x_i^* \rceil$

- Randomized rounding: set $x_i = 1$ with probability x_i^*

$$\mathbb{E}[\sum_{i=1}^{m} x_i] = \sum_{i=1}^{m} \mathbb{E}[x_i] = \sum_{i=1}^{m} x_i^{\ *}$$

- But is it feasible? [Further analysis on board & Panigrahi's notes]

Theorem. There is a poly-time randomized algorithm achieving $O(\log n)$ expected approximation ratio, except w. probability $O(1/n)$.