Wrapping up NP and Intro to Probability

Admin

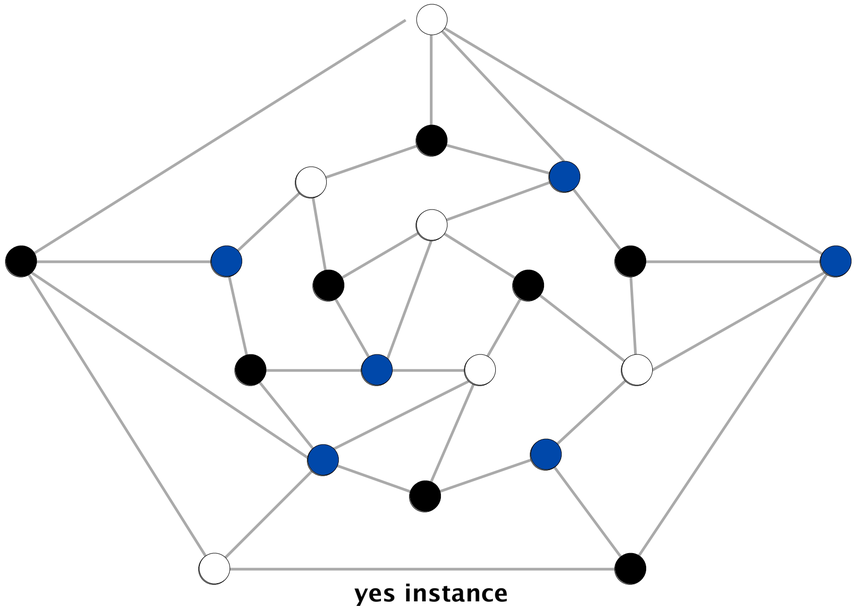
- Move Tai office hours (6-8 Wed) to 6-8 Thur?
    - Email me today if that's an issue
- Assgn 6 back, Assgn 7 out, later today
- Readings on Glow for probability
- Assignment 7 is last before Thanksgiving
- Assignment 8 out Nov 27. Only 9 assignments this semester.
- Attendance not required in class when we go remote
    - Please come anyway if you can!

Graph-3-Color is NP Complete:

3-SAT $\leq_p$ Graph 3-Color

Graph 3-Color Problem

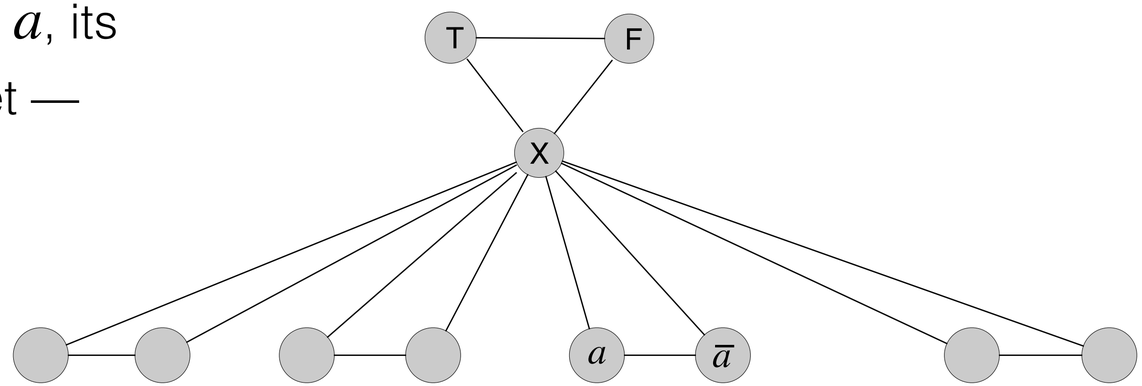
- **3-COLOR**. Given an undirected graph $G = (V, E)$, is it possible to color the vertices with 3 colors s.t. no adjacent nodes have the same color.
- We argued previously that **3-COLOR** $\in$ NP.



yes instance

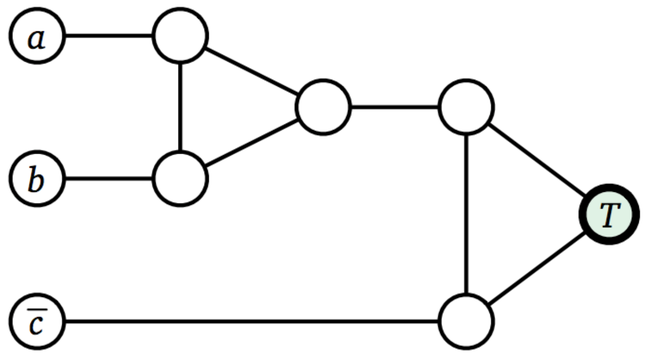
3-SAT to 3-Color Problem

- **Theorem.** **3-SAT** $\leq_p$ **3-COLOR**
- **Proof.** Given a 3-SAT instance Φ, define G as follows
  - Truth gadget: a triangle with three nodes $T, F,$ and X (for true, false and other) — they must get different colors (say *true*, *false*, *other*)
  - Variable gadget: triangle made up of variable a, its negation $\overline{a}$ and the X node of the truth gadget — enforces $a, \overline{a}$ are colored true/false



3-SAT to 3-Color Problem

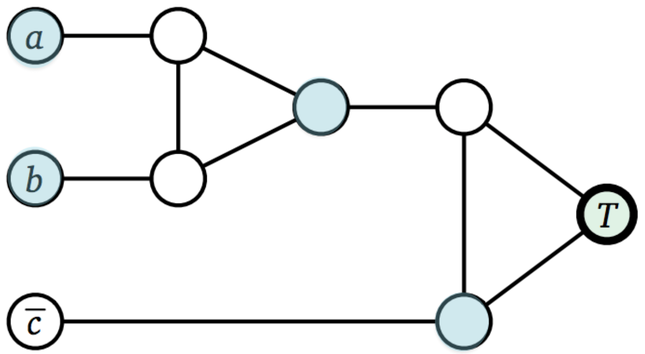
- **Theorem.** **3-SAT $\leq_p$ 3-COLOR**
- **Proof.** Given a 3-SAT instance Φ, define G as follows
  - Truth gadget: a triangle with three nodes $T, F,$ and X (for true, false and other) — they must get different colors (say *true, false, other*)
  - Variable gadget: triangle made up of variable a, its negation $\overline{a}$ and the X node of the truth gadget — enforces $a, \overline{a}$ are colored true/false
  - Clause gadget: joins three literal nodes (from the variable gadget) to node T in the truth gadget using a subgraph as shown below



$(a \vee b \vee \overline{c})$

3-SAT to 3-Color Problem

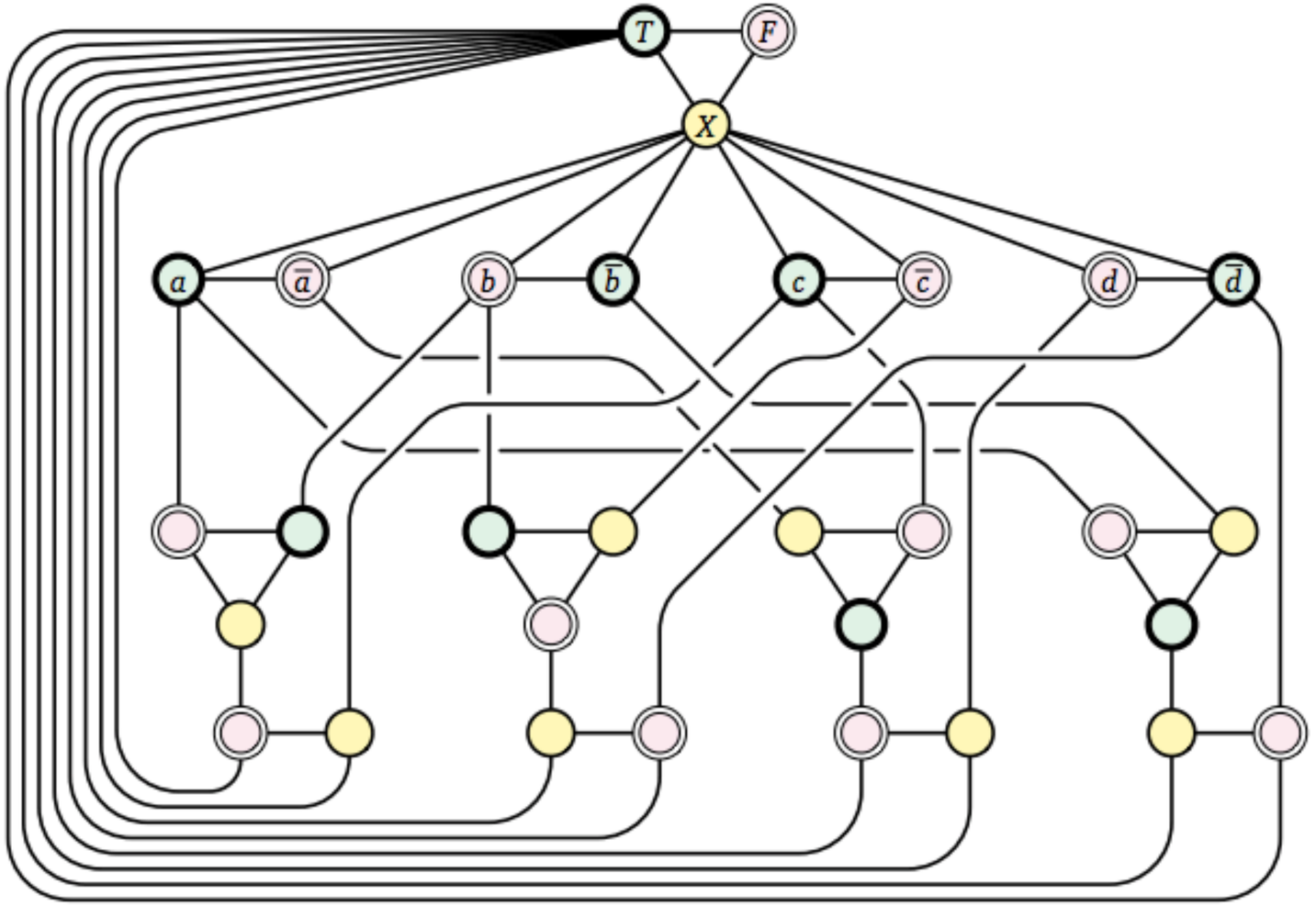
- **Observation**.
  - Clause gadget enforces that in a valid 3-coloring, not all three literals can be colored FALSE
  - If a, b (or $b, \overline{c}$) or ($a, \overline{c}$) get the same color (say, FALSE) then the right-end-point of the triangle must be colored the same (shown in blue)
    - The remaining literal cannot be colored false!



$(a \vee b \vee \overline{c})$

3-SAT to 3-Color Problem

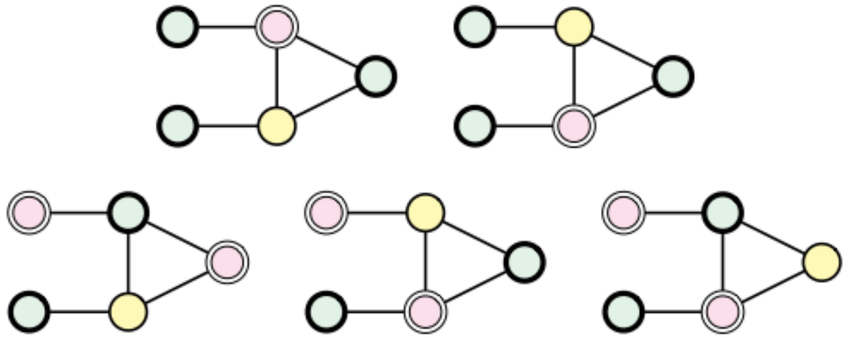
- **Theorem.** 3-SAT $\leq_p$ 3-COLOR
- Overall G example
- (Yes, this is a complicated graph. Complicated graphs are going to be the hard graphs for problems like 3-color!)



$(a \vee b \vee c) \wedge (b \vee \bar{c} \vee \bar{d}) \wedge (\bar{a} \vee c \vee d) \wedge (a \vee \bar{b} \vee \bar{d})$

3-SAT to 3-Color Problem

- **Theorem.** 3-SAT $\leq_p$ 3-COLOR
- **Proof Sketch**.
  - ($\Rightarrow$) If Φ is satisfiable, color the variables based on the satisfying assignment (and because each clause is satisfied) extend the coloring to the clause gadgets
  - ($\Leftarrow$) If G is 3-colorable, then we can assign truth values based on the colors (at least one of the literals in each clause must be colored true) and thus the resulting assignment must satisfy Φ
- Note this problem extends to k-coloring of graphs for $k \geq 3$ and the generalized problem is also hard.



List of NPC Problems So Far

- SAT/ 3-SAT
- INDEPENDENT SET
- VERTEX COVER
- SET COVER
- CLIQUE
- 3-COLOR
- Subset-Sum
- Knapsack
- Next:
  - Traveling salesman problem
  - Hamiltonian cycle / path

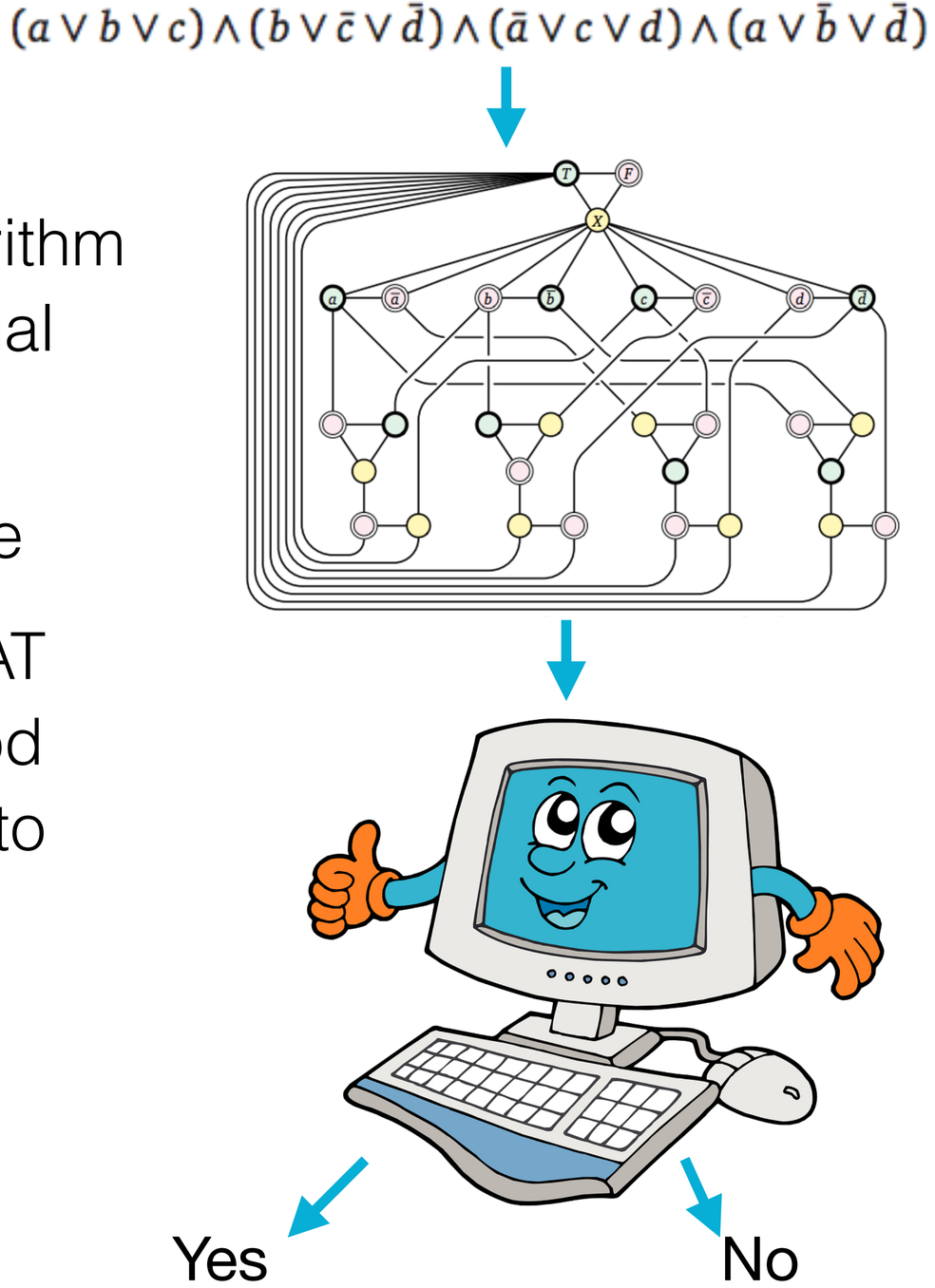
NP Hardness

- Another explanation about what we're proving and why.

    - What do you start with an instance of?
    - What information do you have?
    - What do you need to prove?

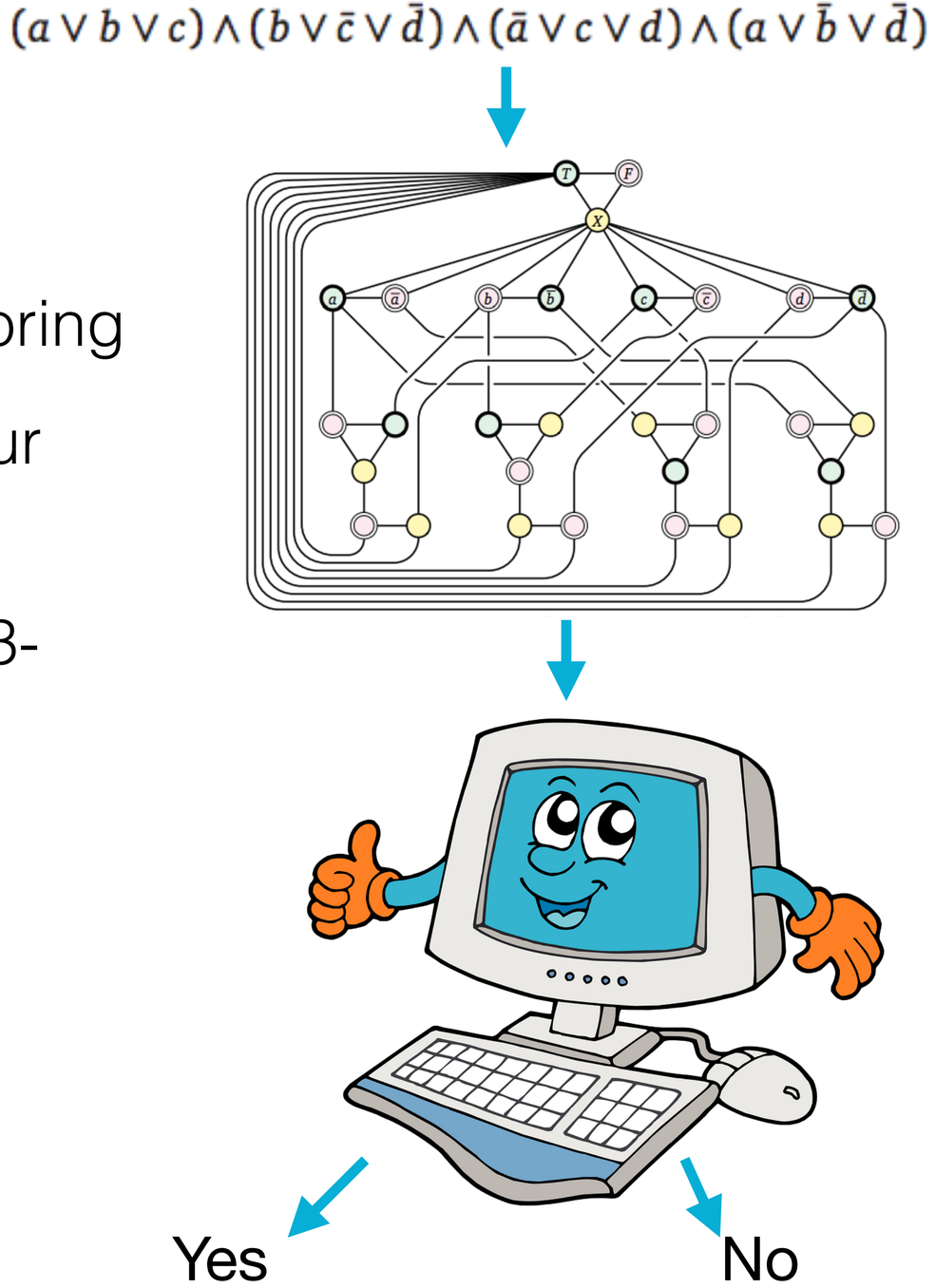
NP Hardness

- Let's say you told me you had an algorithm that could solve 3-coloring in polynomial time
- Then I come to you with a SAT instance
- In polynomial time we transform the SAT instance into a graph (using the method from last slide), and feed that graph into your algorithm

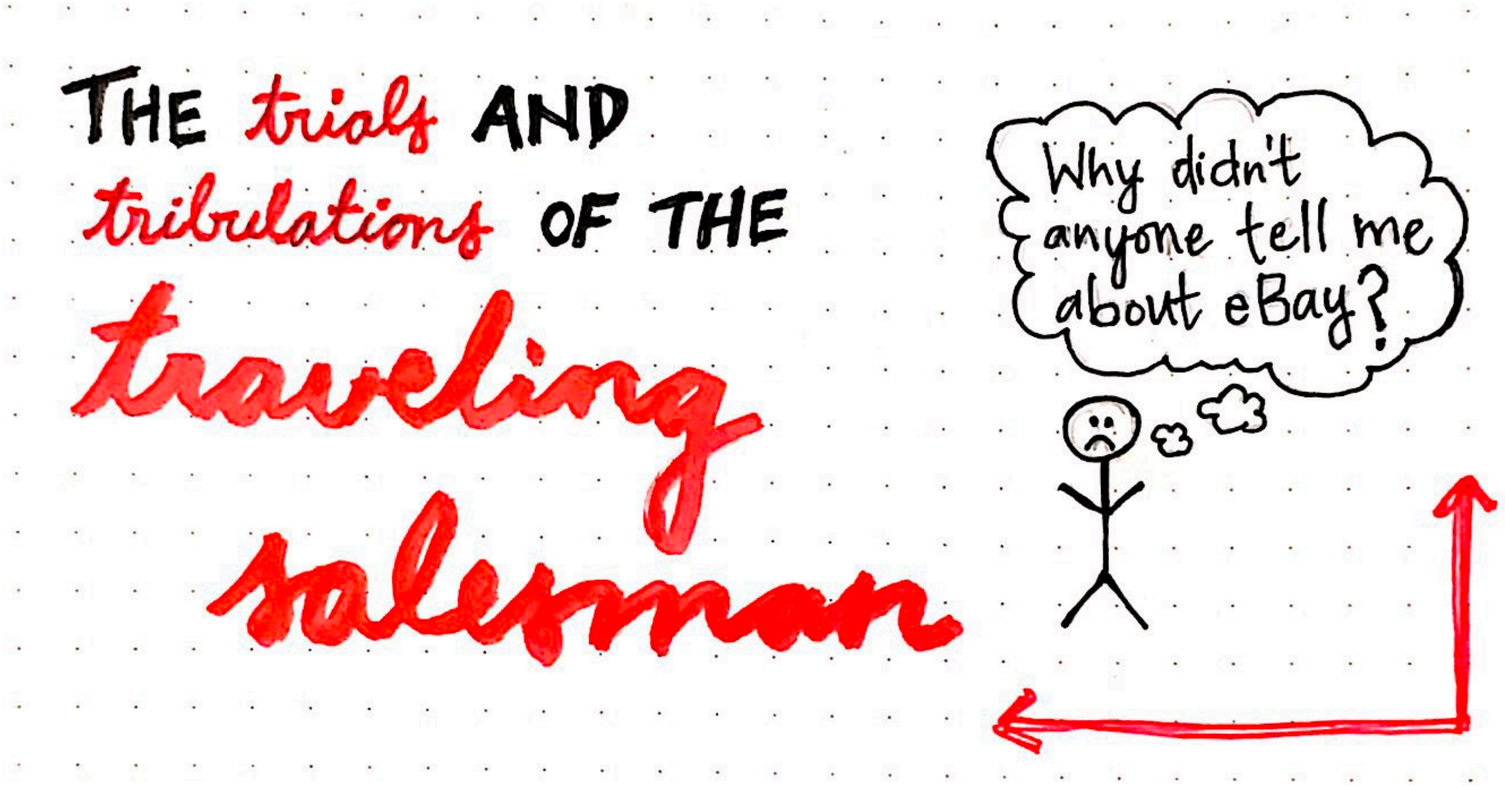


NP Hardness

- (We proved) If my SAT instance is satisfiable, your algorithm finds a 3-coloring
- If my SAT instance is not satisfiable, your algorithm does not find a 3-coloring
    - Same as: if your algorithm finds a 3-coloring, my SAT instance must be satisfiable
- Contradiction! SAT (probably) can't be solved in polynomial time



Traveling Salesman Problem



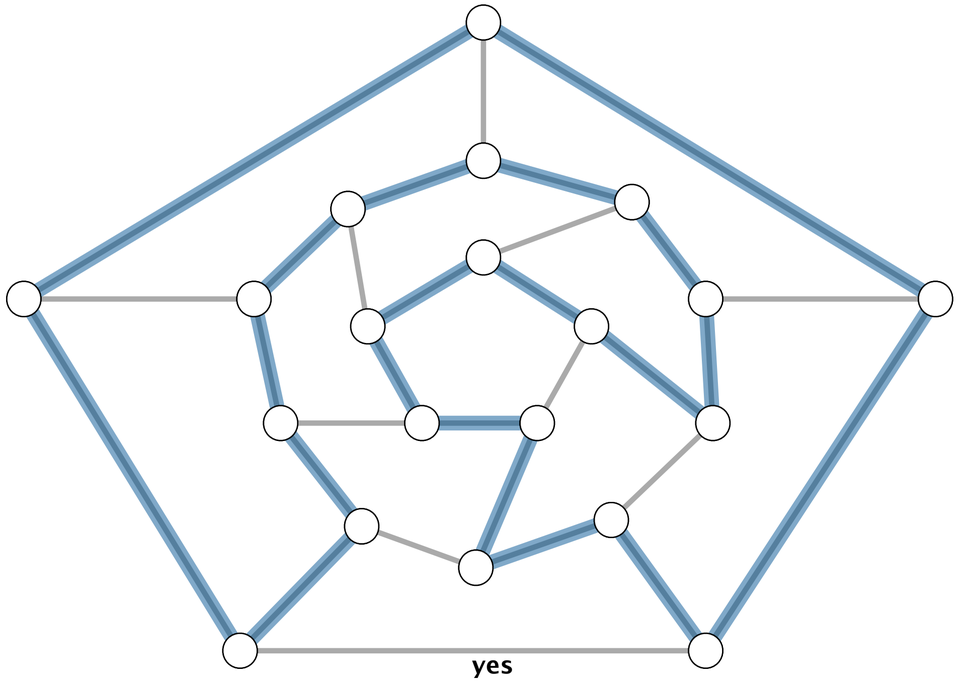
Vaidehi Joshi
https://medium.com/basecs/the-trials-and-tribulations-of-the-traveling-salesman-56048d6709d

Traveling Salesman Problem

- Extremely famous NP complete problem
- Consider a salesman who visits n cities labeled $v_1, \ldots, v_n$
- Salesman starts at v_1 and wants to find a *tour,* an order in which to visit all the other cities and return home
- **Goal**. Travel as little distance as possible
- Formally, let $d(i, j)$ be the distance from city v_i to city v_j (not necessarily symmetric or triangle inequality (e.g. airplane prices))
- **TSP.** Decision version: given a set of distances on n cities and a bound D, is there a tour (of all the cities) of length at most D?
- Many applications: VLSI design, robotics, cache-efficiency
- Will prove **TSP** is NP hard using a similar problem: **HAMILTONIAN CYCLE/PATH**

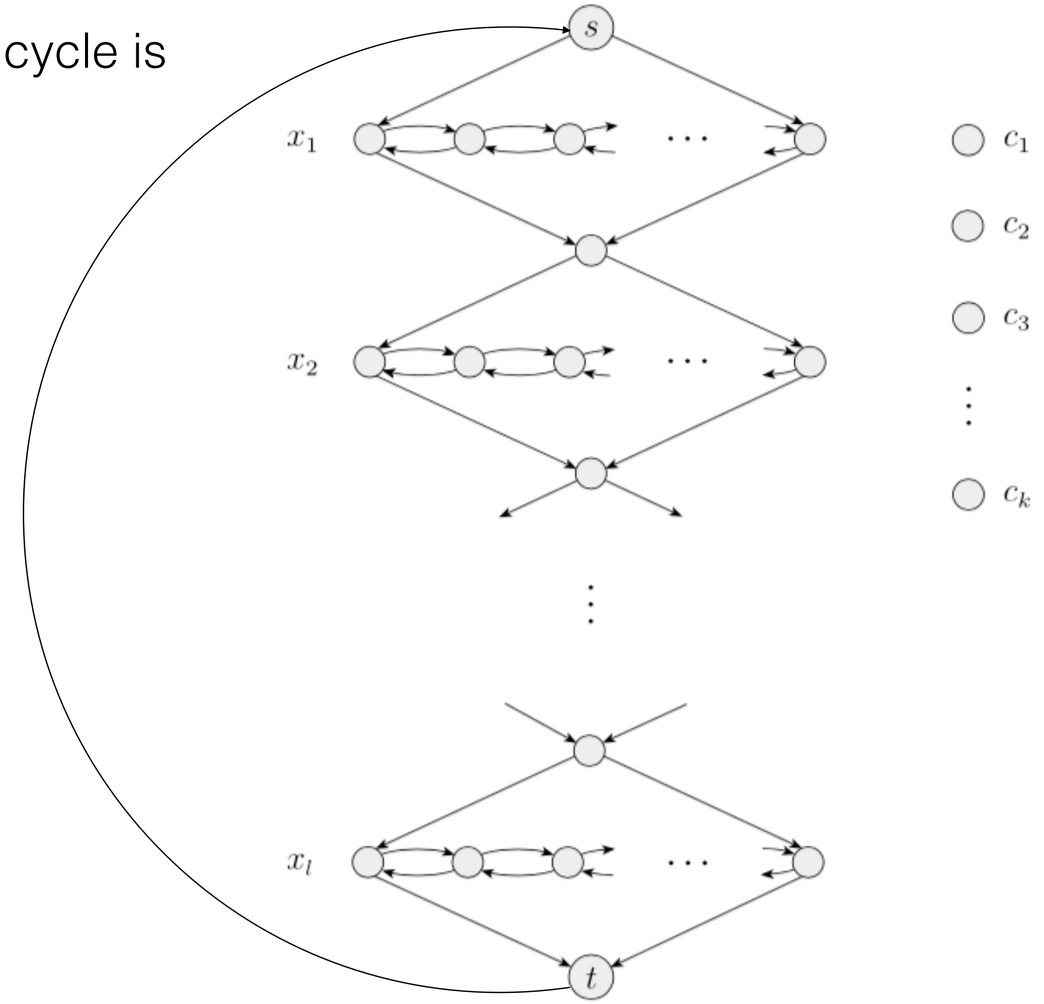
(Directed) Hamiltonian Cycle

- **HAMILTONIAN-CYCLE.** Given a directed graph $G = (V, E)$ does there exists a cycle T that visits every vertex exactly once?
- We want to prove **HAMILTONIAN-CYCLE** is NP complete
  - **HAMILTONIAN-CYCLE** $\in$ NP
    - Certificate: sequence of vertices in the graph
    - Poly-time verifier
      - Check if sequence is a valid path in G
      - Check if path visits every vertex exactly once
  - **HAMILTONIAN-CYCLE is NP hard**
    - Sufficiently different from other NP hard graph problems
    - We (won't) reduce 3SAT to it



3SAT $\leq_p$ Hamiltonian Cycle

- This is what the proof looks like. You'll probably see it in 361
- In this class we will take it as given that Hamiltonian cycle is NP-hard.



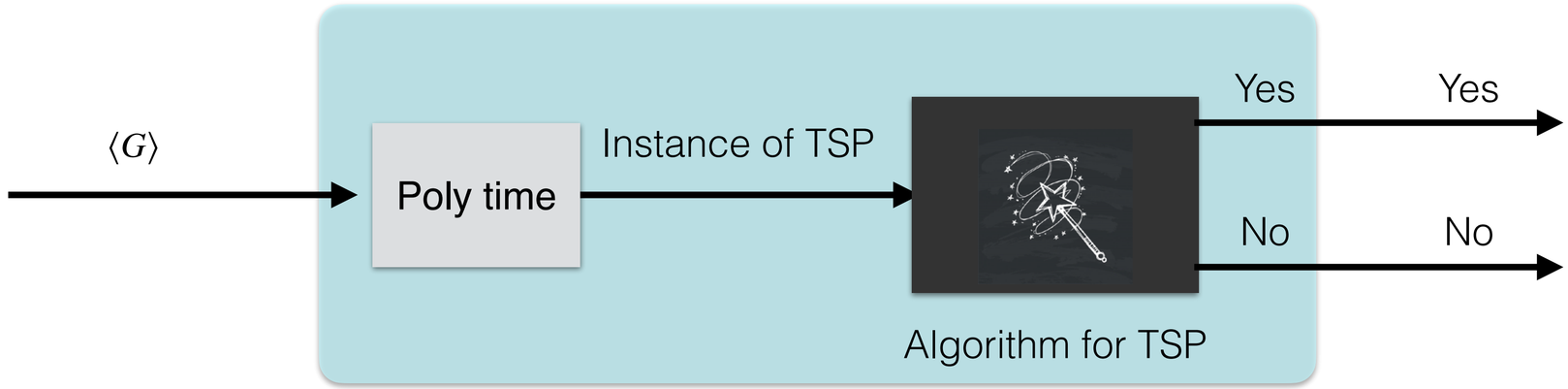
Hamiltonian-Cycle $\leq_p$ TSP

Hamiltonian Cycle to TSP

In Class Exercise. **HAMILTONIAN-CYCLE $\leq_p$ TSP**

Given a directed graph G, convert it to an instance of TSP: that is,

- Cities $c_1, \ldots, c_n$
- $d(i, j)$: distance from city i to city j (all pairs of cities need a distance)
- Target D such that G has a hamiltonian cycle iff there exists a tour of n cities of length at most D



Algorithm for HAM CYCLE

TSP is NP Complete

- **Claim.** **TSP** $\in$ NP
- **Claim.** **HAMILTONIAN-CYCLE** $\leq_p$ **TSP**
- **Proof.** Given directed graph $G = (V, E)$, define instance of TSP as:
  - City v_i' for each node v_i
  - $d(i', j') = 1$ if $(v_i, v_j) \in E$
  - $d(i', j') = 2$ if $(v_i, v_j) \notin E$
- G has a Hamiltonian cycle iff there is a tour of length at most n
- $(\Rightarrow)$ If G has a hamiltonian cycle, then it defines a tour of length n
- $(\Leftarrow)$ Suppose there is a tour of length at most n, why does this ordering correspond to a hamiltonian cycle?

(Directed) Hamiltonian Path

- **HAMILTONIAN-PATH.** Given a directed graph $G = (V, E)$ does there exists a path P that visits every vertex exactly once? Such a path is called a hamiltonian path
- Note: path is allowed to start and end anywhere as long as it visits every node exactly once
- **HAMILTONIAN-PATH** $\in$ NP
  - Certificate: path in G
  - Verifier: check if path visits each node exactly once
- To prove **HAMILTONIAN PATH** is NP hard, we can either
  - We can modify our hamiltonian cycle reduction (delete $t \rightarrow s$)
  - More fun: (exercise) Directly reduce from **HAMILTONIAN CYCLE**

Fun Facts

- Hamiltonian path problem says NP complete even on undirected, two connected, cubic and planar graphs!
- Still NP complete on general grid graphs, but poly-time solvable on “solid grid graphs” (a Williams undergrad thesis by Chris Umans)

SIAM J. COMPUT.
Vol. 5, No. 4, December 1976

THE PLANAR HAMILTONIAN CIRCUIT PROBLEM IS NP-COMPLETE*

M. R. GAREY†, D. S. JOHNSON† AND R. ENDRE TARJAN‡

Abstract. We consider the problem of determining whether a planar, cubic, triply-connected graph *G* has a Hamiltonian circuit. We show that this problem is NP-complete. Hence the Hamiltonian circuit problem for this class of graphs, or any larger class containing all such graphs, is probably computationally intractable.

Key words. algorithms, computational complexity, graph theory, Hamiltonian circuit, NP-completeness

1. Introduction. A *Hamiltonian circuit* in a graph[1] is a path which passes through every vertex exactly once and returns to its starting point. Many attempts have been made to characterize the graphs which contain Hamiltonian circuits (see [2, Chap. 10] for a survey). While providing characterizations in various special cases, none of these results has led to an efficient algorithm for identifying such graphs in general. In fact, recent results [5] showing this problem to be “NP-complete” indicate that no simple, computationally-oriented characterization is possible. For this reason, attention has shifted to special cases with more restricted structure for which such a characterization may still be possible. One special case of particular interest is that of planar graphs. In 1880 Tait made a famous conjecture [8] that every cubic, triply-connected, planar graph contains a Hamiltonian circuit. Though this conjecture received considerable attention (if true it would have resolved the “four color conjecture”), it was not until 1946 that Tutte constructed the first counterexample [9]. We shall show that, not only do these highly-restricted planar graphs occasionally fail to contain a Hamiltonian circuit, but it is probably impossible to give an efficient algorithm which distinguishes those that do from those that do not.

2. Proof of result. Our proof of this result is based on the recently developed theory of “NP-complete problems”. This class of problems possesses the following important properties:

Hamiltonian Cycles in Solid Grid Graphs
(Extended Abstract)

Christopher Umans*
Computer Science Division
U.C. Berkeley
umans@cs.berkeley.edu

William Lenhart
Computer Science Department
Williams College
lenhart@cs.williams.edu

Abstract

A grid graph is a finite node-induced subgraph of the infinite two-dimensional integer grid. A solid grid graph *is a grid graph without holes. For general grid graphs, the Hamiltonian cycle problem is known to be $\mathcal{NP}$-complete. We give a polynomial-time algorithm for the Hamiltonian cycle problem in solid grid graphs, resolving a longstanding open question posed in [IPS82]. In fact, our algorithm can identify Hamiltonian cycles in* quad-quad graphs, *a class of graphs that properly includes solid grid graphs.*

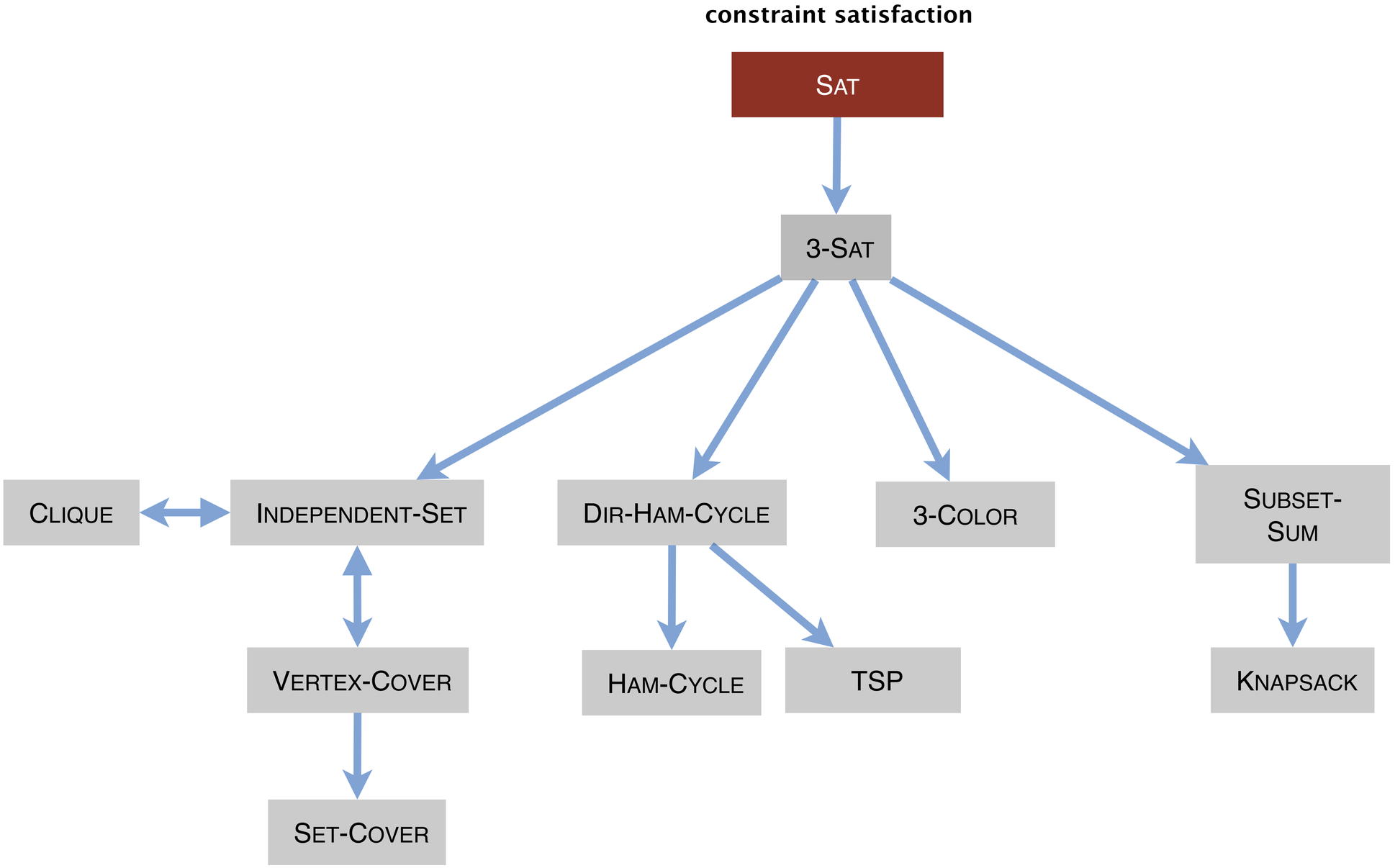
1 Introduction

A *grid graph* is a finite node-induced subgraph of the infinite two-dimensional integer grid. A *solid grid graph* is a grid graph all of whose bounded faces have area one. The study of Hamiltonian cycles in grid graphs was initiated by Itai, Papadimitriou and Szwarcfiter [IPS82], who proved that the problem for general grid graphs is $\mathcal{NP}$-complete, and gave a polynomial-time algorithm for rectangular solid grid graphs. The question of whether a polynomial-time

trails (a relaxation of Hamiltonian cycles) in a broad subclass of grid graphs called *polymino*, have even conjectured that for solid grid graphs, deciding Hamiltonicity is $\mathcal{NP}$-complete.

We present a polynomial-time algorithm that finds Hamiltonian cycles in solid grid graphs using the well-known technique of *cycle merging*. Given an input graph *G*, we first find a *2-factor*, which is a spanning subgraph for which all vertices have degree two. The 2-factor is a set of disjoint cycles that exactly cover the vertices of *G*; a Hamiltonian cycle is a 2-factor with a single component. We then repeatedly identify a transformation of the 2-factor that reduces the number of components. This process either identifies a Hamiltonian cycle or terminates with multiple components if one does not exist.

Our algorithm can be applied to a generalization of solid grid graphs which are “locally” solid grid graphs but may not be fully embeddable in the integer grid without overlap. We call these graphs *quad-quad*


constraint satisfaction
SAT
3-SAT
CLIQUE
INDEPENDENT-SET
DIR-HAM-CYCLE
3-COLOR
SUBSET-SUM
VERTEX-COVER
HAM-CYCLE
TSP
KNAPSACK
SET-COVER

Intro to Probability

Why Probability?

- Randomization is extremely useful in algorithms
    - Quicksort
    - Hashing
    - Simple linear-time median finding
    - We'll see others

- Plan: we'll start with some things you've likely seen before.
  But I want you to have a good foundation.

“Deathbed” Formulas

- You should remember these even on your deathbed
- *Extremely* useful in probability
- $\left(1+\frac{1}{n}\right)^n \approx e \qquad \left(1-\frac{1}{n}\right)^n \approx \frac{1}{e}$ for large enough n (gets close quite quickly)
- More precisely: $\left(1+\frac{1}{n}\right)^n \leq e \qquad \left(1-\frac{1}{n}\right)^n \leq \frac{1}{e}$
- $\left(\frac{x}{y}\right)^y \leq \binom{x}{y} \leq \left(\frac{ex}{y}\right)^y$

$\binom{x}{y} = \frac{x!}{y!(x-y)!}$ is the number of

y-sized subsets of x items

Discrete Probability Review

- A discrete probability space consists of a non-empty, countable set Ω, called the *sample space,* and a probability mass function $\Pr : \Omega \to \mathbb{R}$ s.t.
  - $\Pr[\omega] \geq 0 \quad \forall \omega \in \Omega$ and $\sum_{\omega \in \Omega} \Pr[\omega] = 1$
- E.g.
  - A fair coin: $\Omega = \{\text{heads, tails}\}$ and $\Pr[\text{heads}] = \Pr[\text{tails}] = 1/2$
  - A fair six-sided die: $\Omega = \{1,2,3,4,5,6\}$ and $\Pr[\omega] = 1/6 \quad \forall \omega \in \Omega$

Discrete Probability Review

- Idea: the sample space consists of *all possible outcomes*
- If you're stuck on a probability question, sometimes it may help to list all possible outcomes!
- An **event** is a set of outcomes
- The probability of an event is the weight of all outcomes satisfying that event

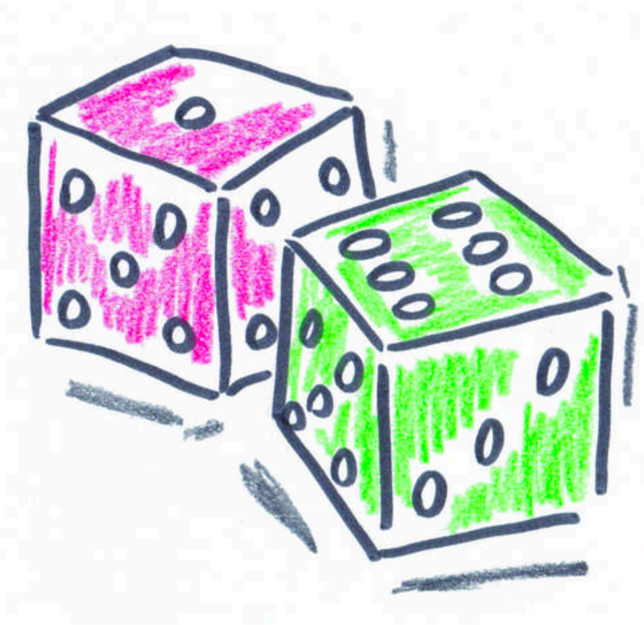
Four Step Method

- Step 1. Find the sample space
- Step 2. Define events of interest
- Step 3. Determine outcome probabilities
- Step 4. Determine event probabilities

When it comes to probability:

Intuition: Bad

Formalism: Good



Example

- Let's say every baby is a girl or a boy with probability $1/2$ each
- If someone has four children, is it more likely that they have two girls and two boys? Or three of one, and one of the other?
- First: what is the sample space/how many outcomes do we have? What is their weight?

Example

- Let's say every baby is a girl or a boy with probability $1/2$ each
- If someone has four children, is it more likely that they have two girls and two boys? Or three of one, and one of the other?
- First: what is the sample space/how many outcomes do we have? What is their weight?

BBBG	GGGG	GGGG	GBBG
BBGB	GGGB	BBBB	BGGB
BGBB	GGBG	GGBB	BGBG
GBBB	GBGG	GBGB	BBGG

Example

- Let's say every baby is a girl or a boy with probability $1/2$ each
- If someone has four children, is it more likely that they have two girls and two boys? Or three of one, and one of the other?

$$8/16 = 1/2$$

BBBG	BGGG	GGGG	GBBG
BBGB	GGGB	BBBB	BGGB
BGBB	GGBG	GGBB	BGBG
GBBB	GBGG	GBGB	BBGG

Example

- Let's say every baby is a girl or a boy with probability $1/2$ each
- If someone has four children, is it more likely that they have two girls and two boys? Or three of one, and one of the other?

$$6/16 < 1/2$$

BBBG	BGGG	GGGG	GBBG
BBGB	GGGB	BBBB	BGGB
BGBB	GGBG	GGBB	BGBG
GBBB	GBGG	GBGB	BBGG

Example (using math)

- Let's say every baby is a girl or a boy with probability $1/2$ each
- If someone has four children, is it more likely that they have two girls and two boys? Or three of one, and one of the other?

- Each outcome occurs with probability $1/2^4$
- $\binom{4}{1} = 4$ ways to have one girl; 4 ways to have one boy; total = $8/16$
- $\binom{4}{2} = 6$ ways to have two girls and two boys; total = $6/16$

Independence

- Intuition: two events are independent if they do not affect each other
    - We'll see a formal definition in a couple slides
- Example: let's say I flip two coins. The event that the first is a head, and the event that the second is a head, are independent.

- Not-independent events: Let's say I flip a coin 10 times. Consider the following two events:
    - Event 1: Flips 1, 2, and 3 are all heads
    - Event 2: Flips 2, 3, and 4 are all heads
- These are not independent. If Event 1 is true, Event 2 is more likely. If Event 1 is false, Event 2 is less likely.

Independent Probabilities

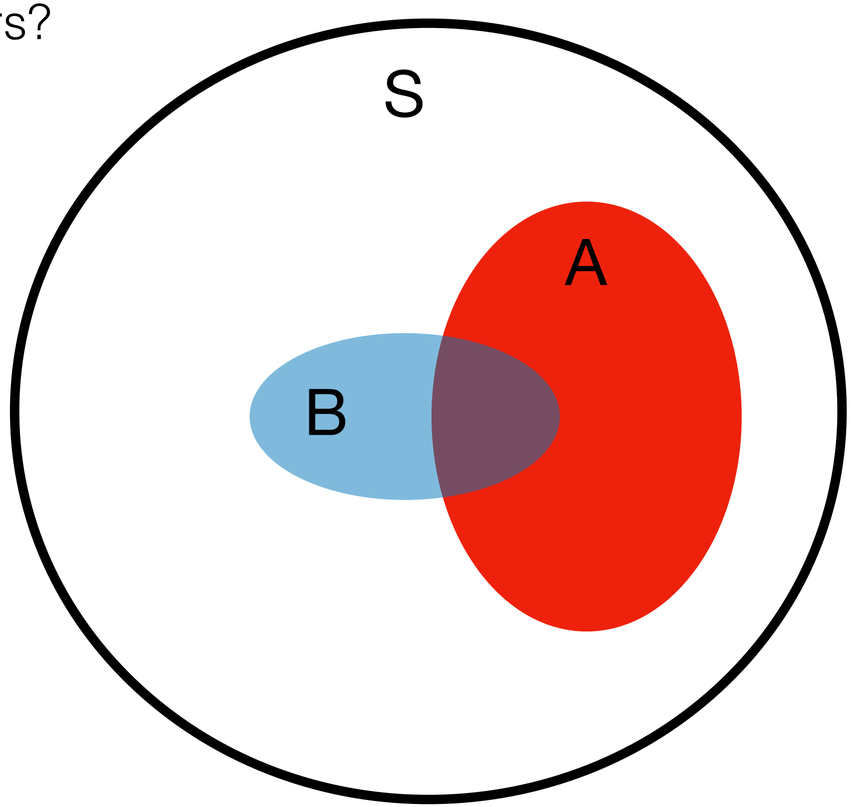
- Definition of independence: A and B are independent events if and only if: $\Pr[A \text{ and } B] = \Pr[A] \cdot \Pr[B]$

- Probability of flipping 10 heads in a row is $1/2^{10}$

- Probability of flipping a heads, and then rolling a 1 on a die, is $1/12$

Conditional Probability

- The probability of event A conditioned on event B is written as $\Pr[A \mid B]$
- Idea: given that B occurred, what is the probability that A occurs?
- Definition: $\Pr[A \mid B] = \dfrac{\Pr[A \text{ and } B]}{\Pr[B]}$
- $\Pr[A]$ is the fraction of S that is red
- $\Pr[A \mid B]$ is the fraction of A that is purple (overlaps with B)



Conditional Probability: Multiplying and Ind.

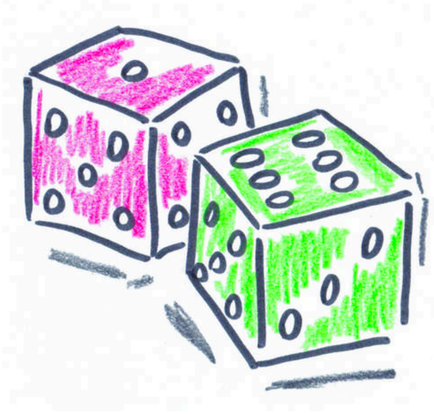
- **Definition of conditional probability:** $\Pr[A \mid B] = \dfrac{\Pr[A \text{ and } B]}{\Pr[B]}$

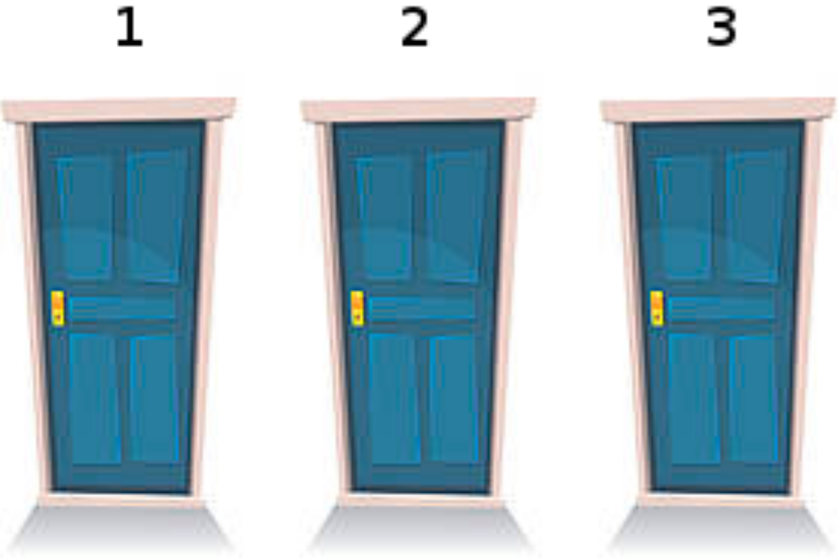
- That means that $\Pr[A \text{ and } B] = \Pr[A \mid B] \cdot \Pr[B]$ (Product rule)

- We know for independent events A and B that $\Pr[A \text{ and } B] = \Pr[A] \cdot \Pr[B]$. So that means that A and B are independent if and only if $\Pr[A \mid B] = \Pr[A]$

Monty Hall Problem

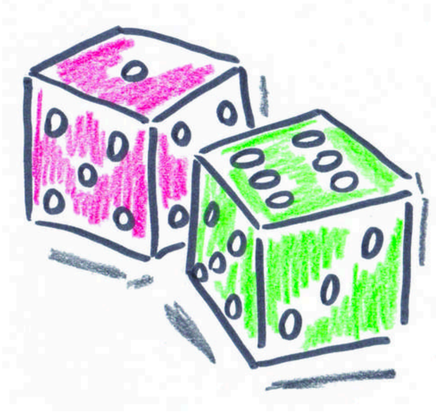
- "*Suppose you're on a game show, and you're given the choice of three doors. Behind one door is a car, behind the others, goats. You pick a door, say number 1, and the host, who knows what's behind the doors, opens another door, say number 3, which has a goat. He says to you, "Do you want to pick door number 2?" Is it to your advantage to switch your choice of doors?*" --- Craig. F. Whitaker Columbia, MD

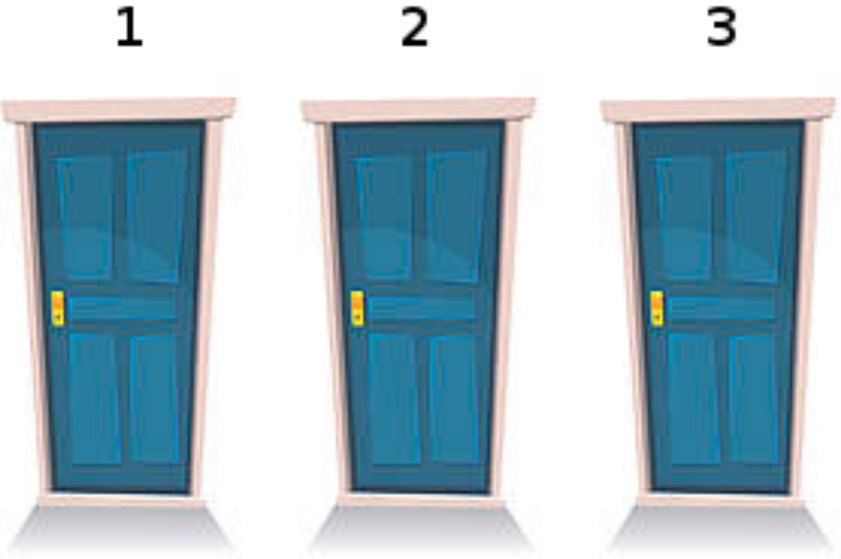




Clarifying the Problem

- The car is equally likely to be hidden behind any of the 3 doors
- The player is equally to pick any of the 3 doors, regardless of the car's location
- After the player picks a door, the host *must* open a *different* door with a goat behind it and offer the choice to switch
- If the host has a choice of which door to open, he is equally likely to select each of them

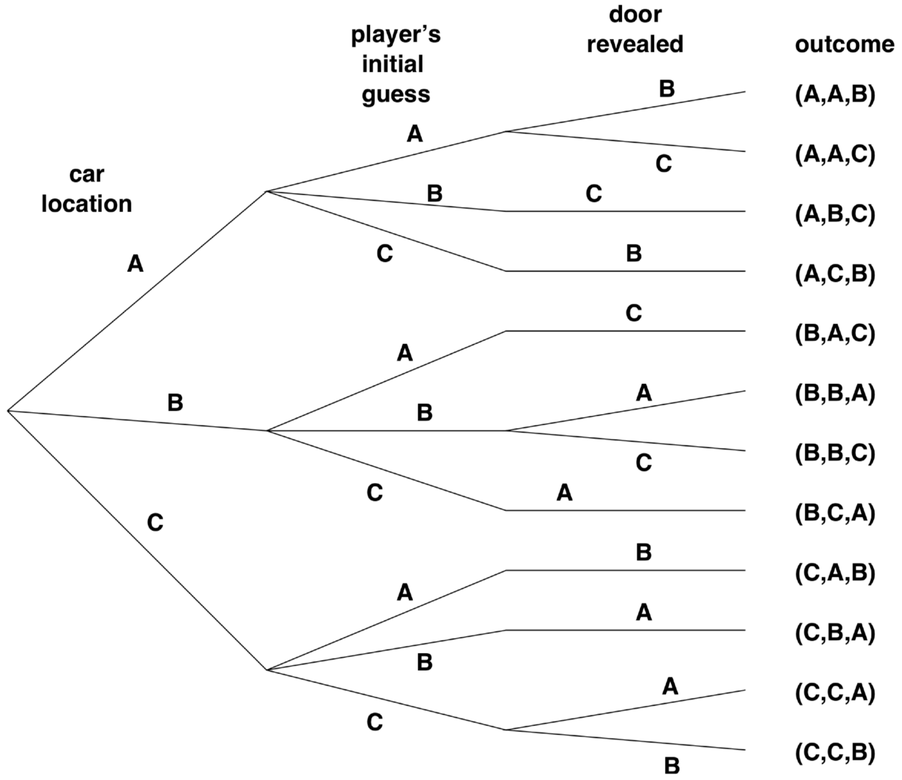




Find the Sample Space

- Sample space: set of all possible outcomes
- An outcome involves 3 things:
  - door concealing the car
  - door initially chosen by the player
  - door that host opens to reveal a goat
- Every possible combination of this is an *outcome*
- We can visualize these as a *tree diagram*
- Sample space S is then:

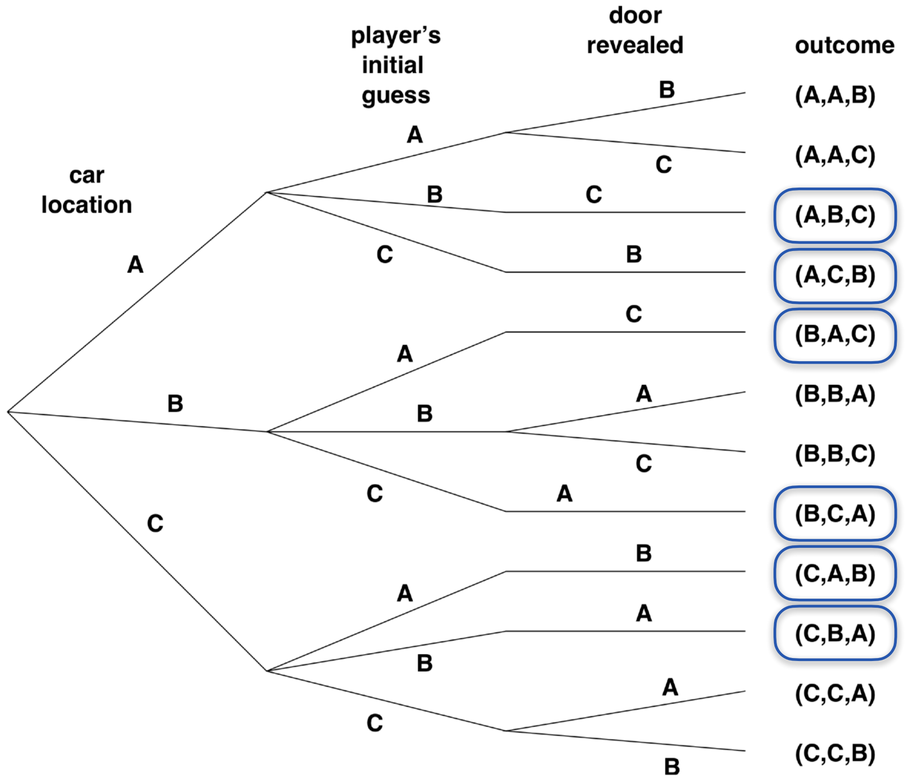
$$S = \left\{ \begin{array}{llllll} (A,A,B), & (A,A,C), & (A,B,C), & (A,C,B), & (B,A,C), & (B,B,A), \\ (B,B,C), & (B,C,A), & (C,A,B), & (C,B,A), & (C,C,A), & (C,C,B) \end{array} \right\}$$



Define Events of Interest

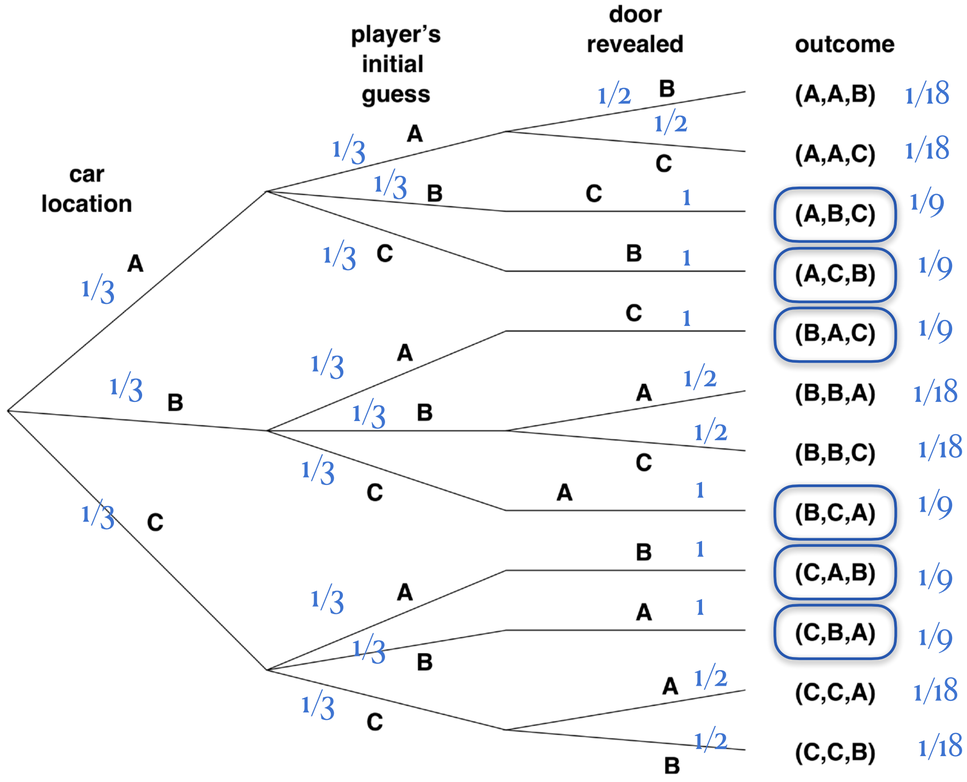
- Question. *What is the probability that _____ ?*
- Model as an ***event*** (subset of the sample space)
- Event that player wins by switching:
  - $\{(A,B,C),(A,C,B),(B,A,C),(B,C,A),(C,A,B),(C,B,A)\}$
  - Exactly half of the outcomes
- Switching leads to win with probability half?
  - No!

$$S = \left\{ \begin{array}{cccccc} (A,A,B), & (A,A,C), & (A,B,C), & (A,C,B), & (B,A,C), & (B,B,A), \\ (B,B,C), & (B,C,A), & (C,A,B), & (C,B,A), & (C,C,A), & (C,C,B) \end{array} \right\}$$



Determine Outcome Probabilities

- Each outcome is not equally likely!
- To determine probability, assign edge probabilities (conditional on previous parts of tree!)
- Pr(A, B, C) $= \frac{1}{18}$, Pr(A, A, C) $= \frac{1}{18}$, Pr(A, B, C) $= \frac{1}{9}$, etc.
- Sum of probabilities of all outcomes is 1
- (Notice) probability is just a function function
  - **Notations.** $\Pr[x]$, $\Pr(x)$
- **Definition** (Probability space)**.**
  A sample space S together with a probability function $\Pr : S \to [0,1]$



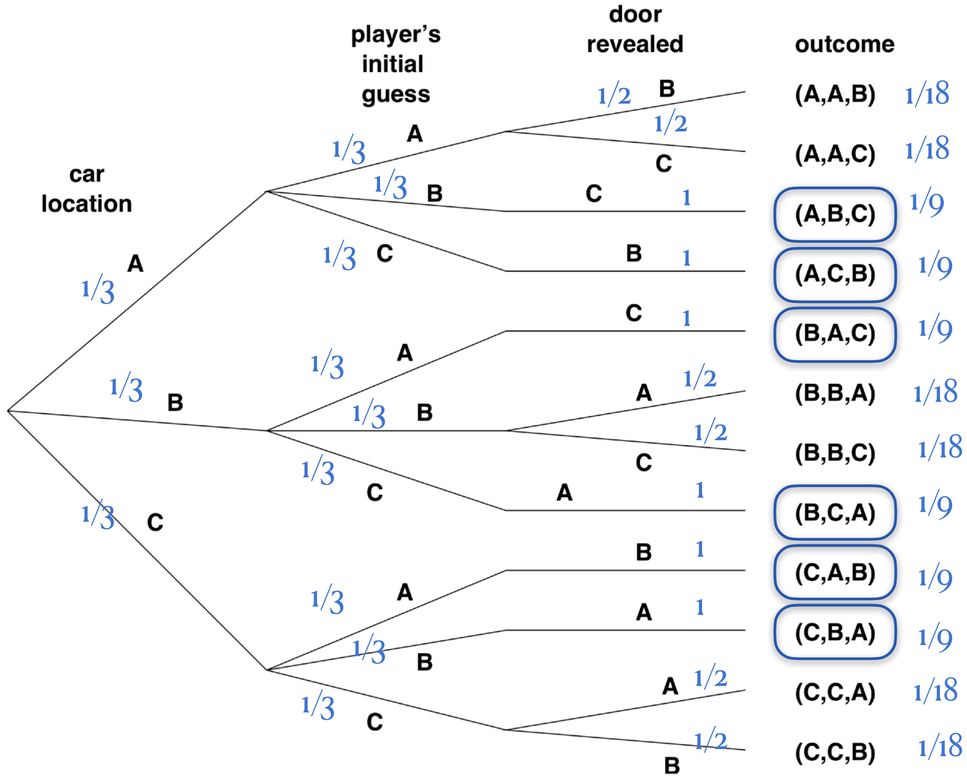
Compute Event Probabilities

- We now have a probability of each outcome
- Probability of an event is the sum of the probabilities of the outcomes it contains, i.e., $\Pr(E) = \sum_{x \in E} \Pr(x)$
- Pr(switching wins) = $\frac{1}{9} + \frac{1}{9} + \frac{1}{9} + \frac{1}{9} + \frac{1}{9} + \frac{1}{9} = \frac{2}{3}$
- It is better to switch!
- Takeaway: resist the intuitively appealing answer

$$S = \left\{ \begin{array}{llllll} (A,A,B), & (A,A,C), & (A,B,C), & (A,C,B), & (B,A,C), & (B,B,A), \\ (B,B,C), & (B,C,A), & (C,A,B), & (C,B,A), & (C,C,A), & (C,C,B) \end{array} \right\}$$

Event (Switching Wins) =

$$\{(A,B,C),(A,C,B),(B,A,C),(B,C,A),(C,A,B),(C,B,A)\}$$



Random Variables

- **Definition.** A random variable X is a function from a sample space S (with a probability measure) to some value set (e.g. real numbers, integers, etc.)

- So for example:
  - I flip a coin 10 times. Let X be the number of heads
  - $\Pr[X = 0] = 1/2^{10}$
  - $\Pr[X = 10] = 1/2^{10}$
  - $\Pr[X = 4]$?
  - $\Pr[X = 4] = \binom{10}{4} \frac{1}{2^4} \frac{1}{2^6} = \frac{105}{512}$

Random Variable

- Event either does or does not happen, what if we want to capture *magnitude* of a probabilistic event
- Suppose I flip n independent fair coins, then the number of heads is a random variable
- Number that comes up when we roll a fair die is a random variable
- If an algorithm flips some coins then the running time of the algorithm is a random variable

- A random variable from S to $\{0,1\}$ is called an *indicator random variable or Bernoulli random variable*

Acknowledgments

- Some of the material in these slides are taken from
  - Kleinberg Tardos Slides by Kevin Wayne (https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/04GreedyAlgorithmsI.pdf)
  - Jeff Erickson's Algorithms Book (http://jeffe.cs.illinois.edu/teaching/algorithms/book/Algorithms-JeffE.pdf)
  - Hamiltonian cycle reduction images from Michael Sipser's Theory of Computation Book