CS 580: Algorithm Design and Analysis

Jeremiah Blocki
Purdue University
Spring 2018

Homework 5 due on March 29th at 11:59 PM (on Blackboard)

Recap

- Polynomial Time Reductions ($X \leq_P Y$)
- P
  - Decision problems which have a polynomial time algorithm.

- NP
  - Decision problems which have a polynomial time *proof certification* algorithm.
    - All YES instances have a short proof
- NP-Hard
  - For *all* $X \in NP$ we have a reduction $X \leq_P Y$
  - A decision problem $Y \in NP$ that is at least as hard any other problem $X \in NP$
- NP-Complete
  - A decision problem $Y \in NP$ that is also NP-Hard.
- We know that $P \subset NP$ and $NP \subset EXP$, but not if $P=NP$

NP-Complete

NP-complete. A problem Y in NP with the property that for every problem X in NP, $X \leq_p Y$.

Theorem. Suppose Y is an NP-complete problem. Then Y is solvable in poly-time iff P = NP.

Pf. $\Leftarrow$ If P = NP then Y can be solved in poly-time since Y is in NP.

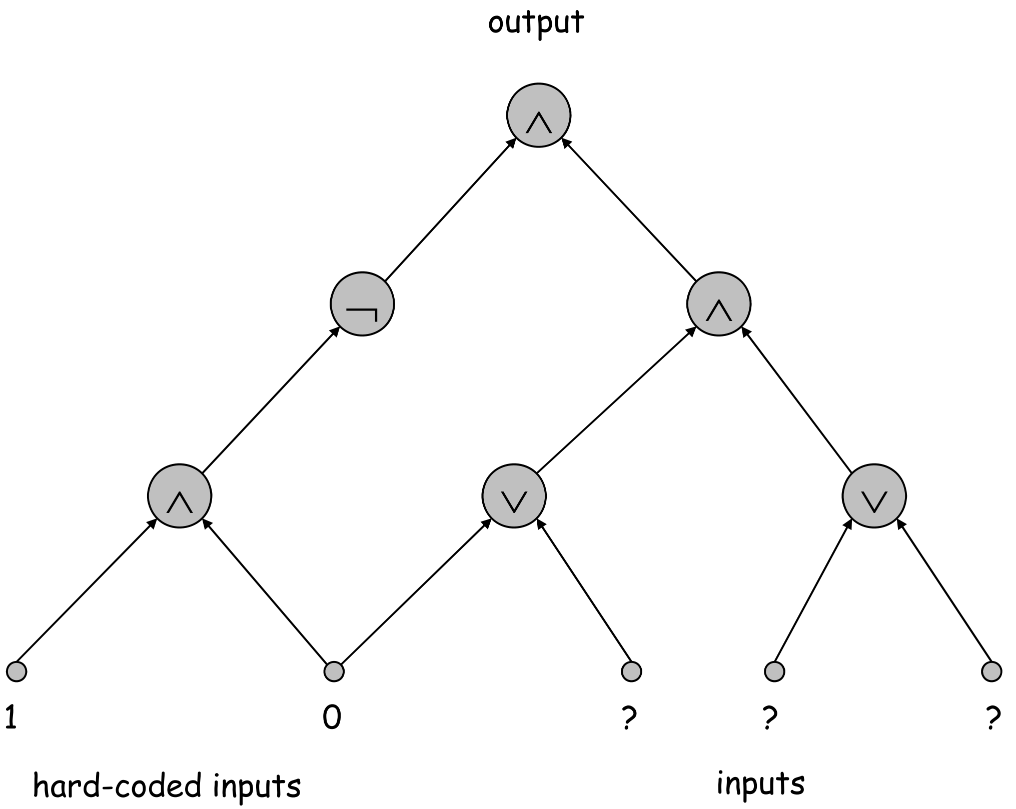
Pf. $\Rightarrow$ Suppose Y can be solved in poly-time.

- Let X be any problem in NP. Since $X \leq_p Y$, we can solve X in poly-time. This implies $NP \subseteq P$.
- We already know $P \subseteq NP$. Thus P = NP. ▪

Fundamental question. Do there exist "natural" NP-complete problems?

Circuit Satisfiability

CIRCUIT-SAT. Given a combinational circuit built out of AND, OR, and NOT gates, is there a way to set the circuit inputs so that the output is 1?



yes: 1 0 1

The "First" NP-Complete Problem

Theorem. CIRCUIT-SAT is NP-complete. [Cook 1971, Levin 1973]

Pf. (sketch)

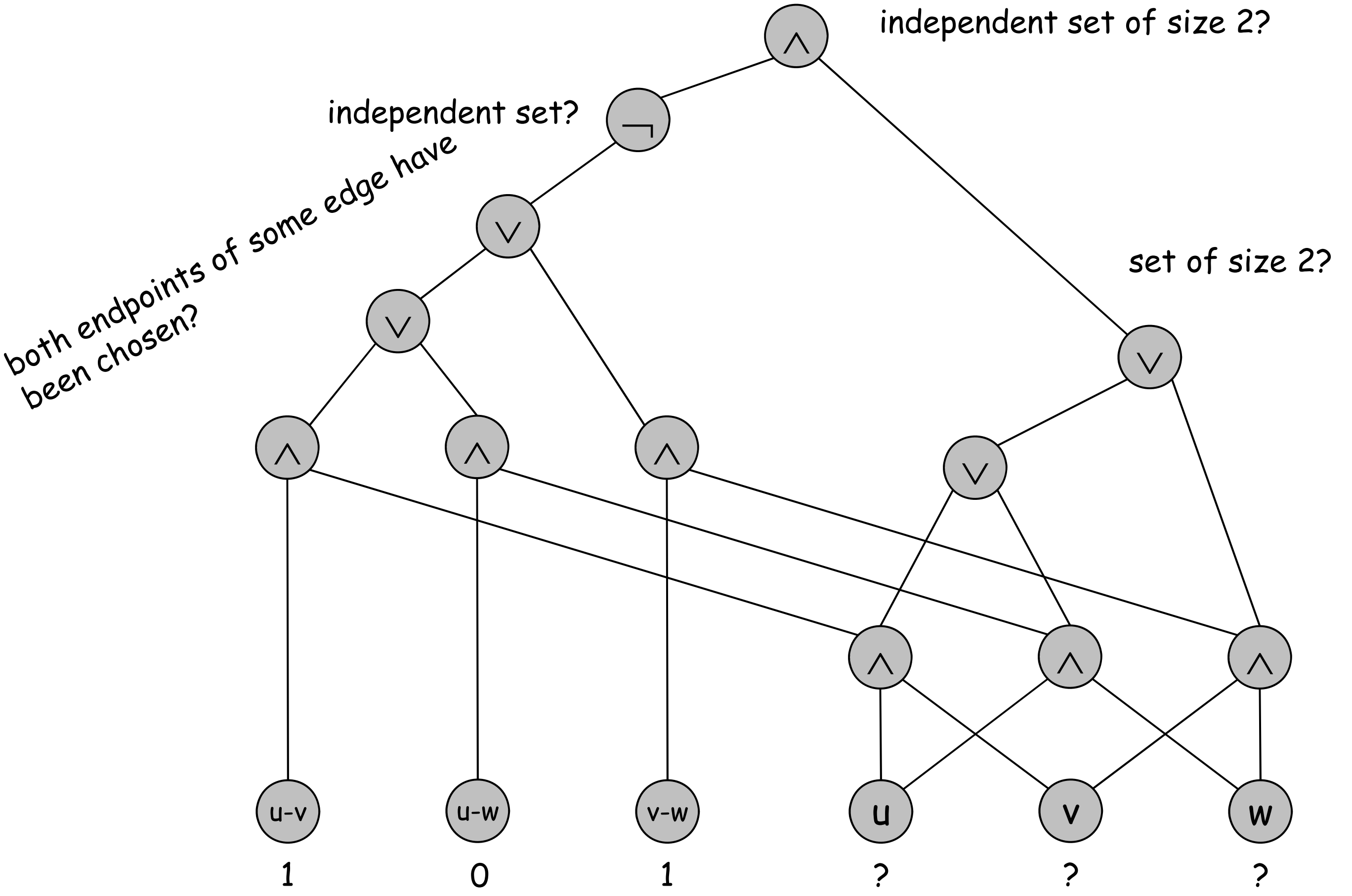
- Any algorithm that takes a fixed number of bits n as input and produces a yes/no answer can be represented by such a circuit. Moreover, if algorithm takes poly-time, then circuit is of poly-size.

  sketchy part of proof; fixing the number of bits is important,
  and reflects basic distinction between algorithms and circuits

- Consider some problem X in NP. It has a poly-time certifier C(s, t). To determine whether s is in X, need to know if there exists a certificate t of length p(|s|) such that C(s, t) = yes.
- View C(s, t) as an algorithm on |s| + p(|s|) bits (input s, certificate t) and convert it into a poly-size circuit K.
  - first |s| bits are hard-coded with s
  - remaining p(|s|) bits represent bits of t
- Circuit K is satisfiable iff C(s, t) = yes.

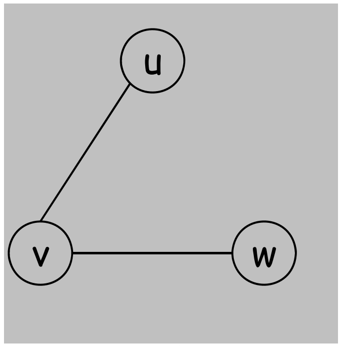
Example

Ex. Construction below creates a circuit K whose inputs can be set so that K outputs true iff graph G has an independent set of size 2.



$\binom{n}{2}$ hard-coded inputs (graph description)

n inputs (nodes in independent set)



G = (V, E), n = 3

Establishing NP-Completeness

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

Recipe to establish NP-completeness of problem Y.

- Step 1. Show that Y is in NP.
- Step 2. Choose an NP-complete problem X.
- Step 3. Prove that $X \le_p Y$.

Justification. If X is an NP-complete problem, and Y is a problem in NP with the property that $X \le_P Y$ then Y is NP-complete.

Pf. Let W be any problem in NP. Then $W \underset{\substack{\uparrow \\ \text{by definition of} \\ \text{NP-complete}}}{\le_P} X \underset{\substack{\uparrow \\ \text{by assumption}}}{\le_P} Y$.

- By transitivity, $W \le_P Y$.
- Hence Y is NP-complete. ▪

3-SAT is NP-Complete

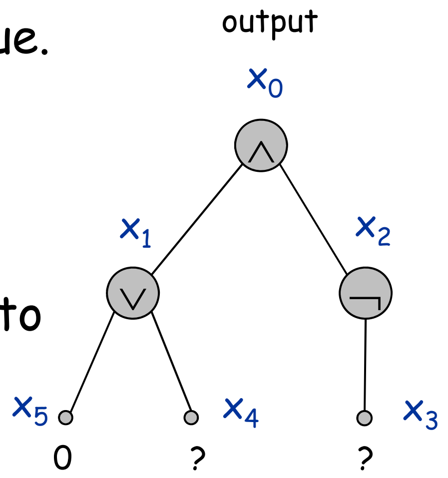
Theorem. 3-SAT is NP-complete.

Pf. Suffices to show that CIRCUIT-SAT $\leq_P$ 3-SAT since 3-SAT is in NP.

- Let K be any circuit.
- Create a 3-SAT variable x_i for each circuit element i.
- Make circuit compute correct values at each node:
  - $x_2 = \neg x_3$ $\Rightarrow$ add 2 clauses: $x_2 \vee x_3$, $\overline{x_2} \vee \overline{x_3}$
  - $x_1 = x_4 \vee x_5$ $\Rightarrow$ add 3 clauses: $x_1 \vee \overline{x_4}$, $x_1 \vee \overline{x_5}$, $\overline{x_1} \vee x_4 \vee x_5$
  - $x_0 = x_1 \wedge x_2$ $\Rightarrow$ add 3 clauses: $\overline{x_0} \vee x_1$, $\overline{x_0} \vee x_2$, $x_0 \vee \overline{x_1} \vee \overline{x_2}$

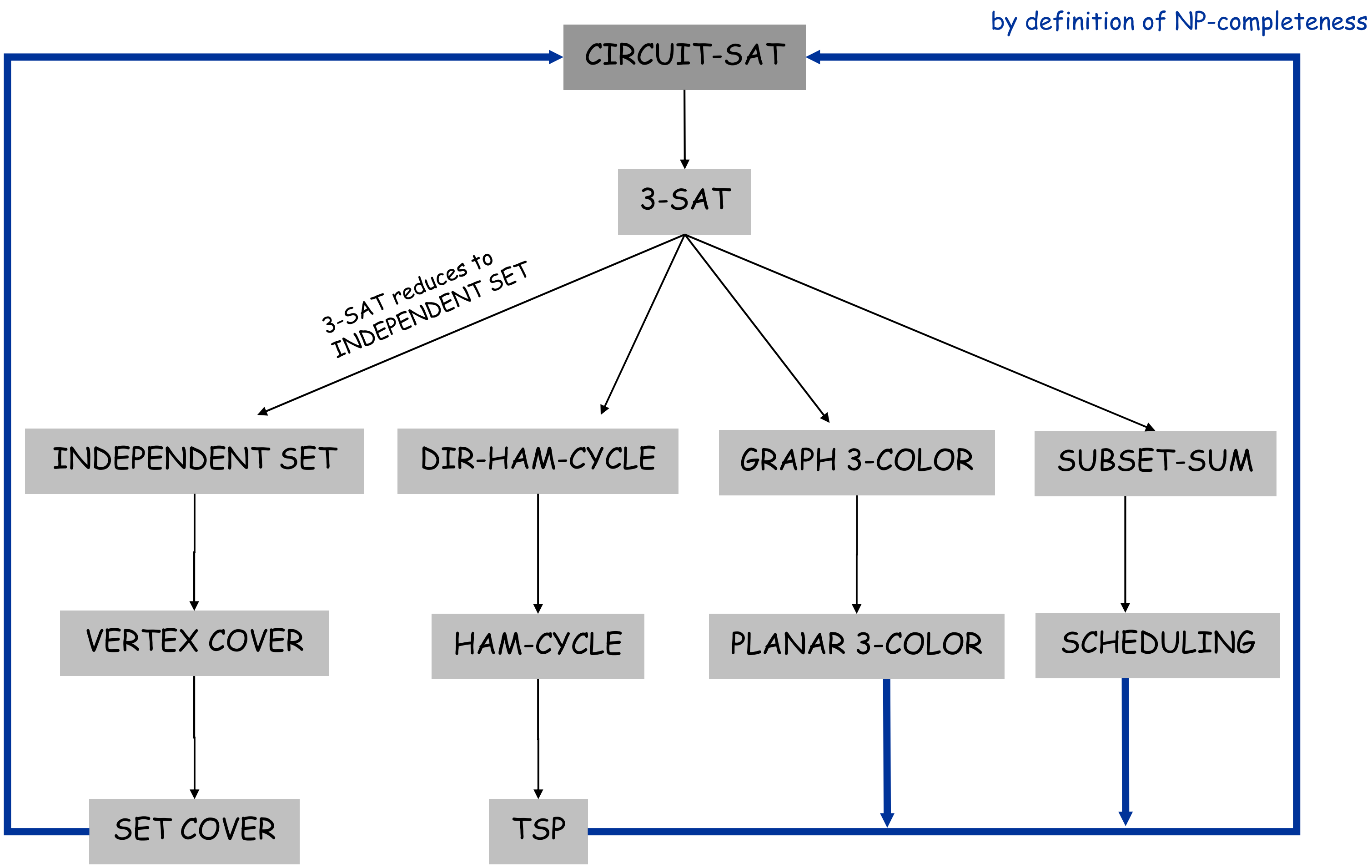
- Hard-coded input values and output value.
  - $x_5 = 0$ $\Rightarrow$ add 1 clause: $\overline{x_5}$
  - $x_0 = 1$ $\Rightarrow$ add 1 clause: x_0

- Final step: turn clauses of length < 3 into clauses of length exactly 3. ▪



NP-Completeness

Observation. All problems below are NP-complete and polynomial reduce to one another!


by definition of NP-completeness

CIRCUIT-SAT

3-SAT

3-SAT reduces to INDEPENDENT SET

INDEPENDENT SET

DIR-HAM-CYCLE

GRAPH 3-COLOR

SUBSET-SUM

VERTEX COVER

HAM-CYCLE

PLANAR 3-COLOR

SCHEDULING

SET COVER

TSP

Some NP-Complete Problems

Six basic genres of NP-complete problems and paradigmatic examples.

- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING, 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.

Practice. Most NP problems are either known to be in P or NP-complete.

Notable exceptions. Factoring, graph isomorphism, Nash equilibrium.

Extent and Impact of NP-Completeness

Extent of NP-completeness. [Papadimitriou 1995]

- Prime intellectual export of CS to other disciplines.
- 6,000 citations per year (title, abstract, keywords).
  - more than "compiler", "operating system", "database"
- Broad applicability and classification power.
- "Captures vast domains of computational, scientific, mathematical endeavors, and seems to roughly delimit what mathematicians and scientists had been aspiring to compute feasibly."

NP-completeness can guide scientific inquiry.

- 1926: Ising introduces simple model for phase transitions.
- 1944: Onsager solves 2D case in tour de force.
- 19xx: Feynman and other top minds seek 3D solution.
- 2000: Istrail proves 3D problem NP-complete.

More Hard Computational Problems

Aerospace engineering: optimal mesh partitioning for finite elements.
Biology: protein folding.
Chemical engineering: heat exchanger network synthesis.
Civil engineering: equilibrium of urban traffic flow.
Economics: computation of arbitrage in financial markets with friction.
Electrical engineering: VLSI layout.
Environmental engineering: optimal placement of contaminant sensors.
Financial engineering: find minimum risk portfolio of given return.
Game theory: find Nash equilibrium that maximizes social welfare.
Genomics: phylogeny reconstruction.
Mechanical engineering: structure of turbulence in sheared flows.
Medicine: reconstructing 3-D shape from biplane angiocardiogram.
Operations research: optimal resource allocation.
Physics: partition function of 3-D Ising model in statistical mechanics.
Politics: Shapley-Shubik voting power.
Pop culture: Minesweeper consistency.
Statistics: optimal experimental design.

8.9 co-NP and the Asymmetry of NP

Asymmetry of NP

Asymmetry of NP. We only need to have short proofs of yes instances.

Ex 1. SAT vs. TAUTOLOGY.

- Can prove a CNF formula is satisfiable by giving such an assignment.
- How could we prove that a formula is not satisfiable?

Ex 2. HAM-CYCLE vs. NO-HAM-CYCLE.

- Can prove a graph is Hamiltonian by giving such a Hamiltonian cycle.
- How could we prove that a graph is not Hamiltonian?

Remark. SAT is NP-complete and SAT $\equiv_P$ TAUTOLOGY, but how do we classify TAUTOLOGY?

↑

not even known to be in NP

NP and co-NP

NP. Decision problems for which there is a poly-time certifier.
Ex. SAT, HAM-CYCLE, COMPOSITES.

Def. Given a decision problem X, its complement $\overline{X}$ is the same problem with the yes and no answers reverse.

Ex. $\overline{X} = \{ 0, 1, 4, 6, 8, 9, 10, 12, 14, 15, \ldots \}$
$X = \{ 2, 3, 5, 7, 11, 13, 17, 23, 29, \ldots \}$

co-NP. Complements of decision problems in NP.
Ex. TAUTOLOGY, NO-HAM-CYCLE, PRIMES.

NP = co-NP ?

Fundamental question. Does NP = co-NP?

- Do yes instances have succinct certificates iff no instances do?
- Consensus opinion: no.

Theorem. If NP ≠ co-NP, then P ≠ NP.

Pf idea.

- P is closed under complementation.
- If P = NP, then NP is closed under complementation.
- In other words, NP = co-NP.
- This is the contrapositive of the theorem.

Good Characterizations

Good characterization. [Edmonds 1965] NP ∩ co-NP.

- If problem X is in both NP and co-NP, then:
  - for yes instance, there is a succinct certificate
  - for no instance, there is a succinct disqualifier
- Provides conceptual leverage for reasoning about a problem.

Ex. Given a bipartite graph, is there a perfect matching.

- If yes, can exhibit a perfect matching.
- If no, can exhibit a set of nodes S such that $|N(S)| < |S|$.

Good Characterizations

Observation. $P \subseteq NP \cap$ co-NP.

- Proof of max-flow min-cut theorem led to stronger result that max-flow and min-cut are in P.
- Sometimes finding a good characterization seems easier than finding an efficient algorithm.

Fundamental open question. Does $P = NP \cap$ co-NP?

- Mixed opinions.
- Many examples where problem found to have a non-trivial good characterization, but only years later discovered to be in P.
  - linear programming [Khachiyan, 1979]
  - primality testing [Agrawal-Kayal-Saxena, 2002]

Fact. Factoring is in $NP \cap$ co-NP, but not known to be in P.

↑

if poly-time algorithm for factoring,
can break RSA cryptosystem

FACTOR is in NP $\cap$ co-NP

FACTORIZE. Given an integer x, find its prime factorization.
FACTOR. Given two integers x and y, does x have a nontrivial factor less than y?

Theorem. FACTOR $\equiv_P$ FACTORIZE.

Theorem. FACTOR is in NP $\cap$ co-NP.
Pf.

- Certificate: a factor p of x that is less than y.
- Disqualifier: the prime factorization of x (where each prime factor is less than y), along with a certificate that each factor is prime.

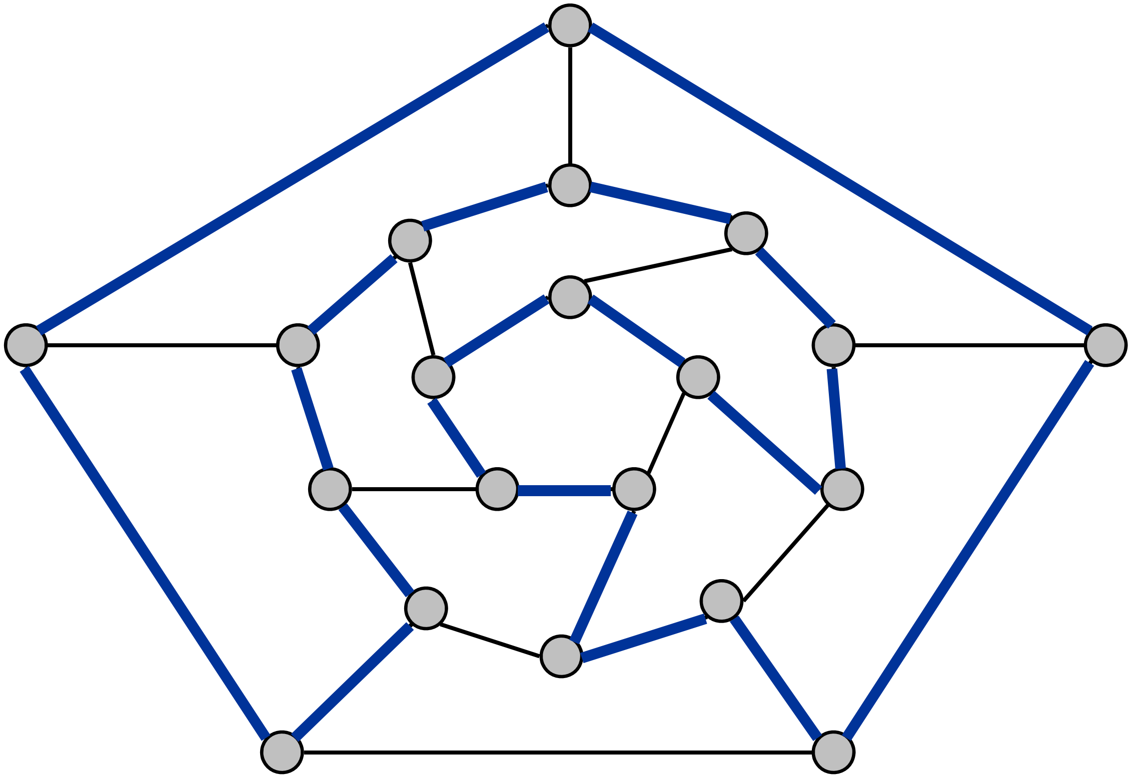
8.5 Sequencing Problems

Basic genres.

- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING, 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.

Hamiltonian Cycle

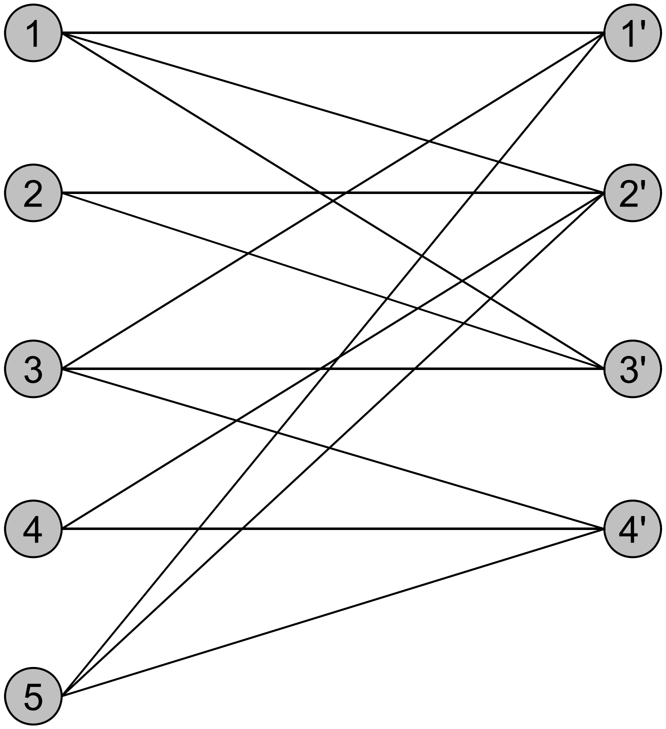
HAM-CYCLE: given an undirected graph G = (V, E), does there exist a simple cycle Γ that contains every node in V.



YES: vertices and faces of a dodecahedron.

Hamiltonian Cycle

HAM-CYCLE: given an undirected graph $G = (V, E)$, does there exist a simple cycle Γ that contains every node in V.



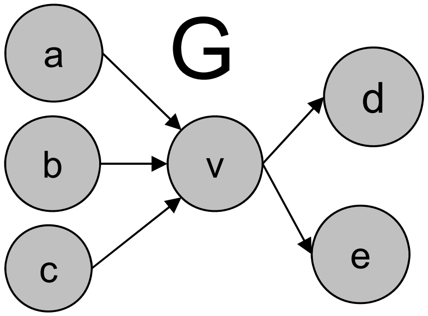
NO: bipartite graph with odd number of nodes.

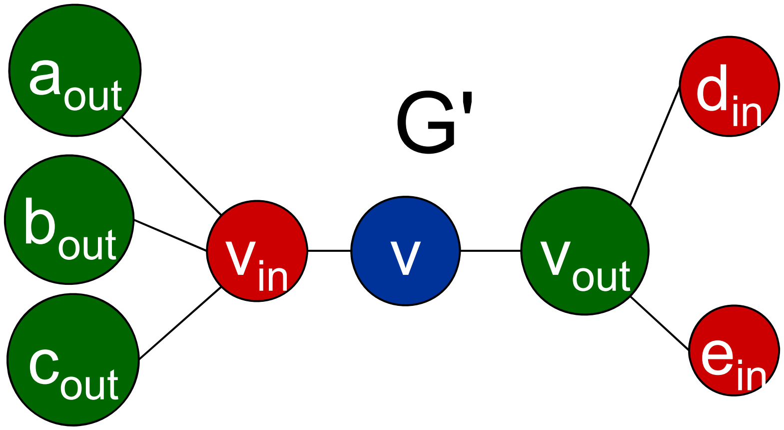
Directed Hamiltonian Cycle

DIR-HAM-CYCLE: given a digraph G = (V, E), does there exists a simple directed cycle Γ that contains every node in V?

Claim. DIR-HAM-CYCLE $\leq_P$ HAM-CYCLE.

Pf. Given a directed graph G = (V, E), construct an undirected graph G' with 3n nodes.





Directed Hamiltonian Cycle

Claim. G has a Hamiltonian cycle iff G' does.

Pf. $\Rightarrow$

- Suppose G has a directed Hamiltonian cycle Γ.
- Then G' has an undirected Hamiltonian cycle (same order).

Pf. $\Leftarrow$

- Suppose G' has an undirected Hamiltonian cycle Γ'.
- Γ' must visit nodes in G' using one of following two orders:
  - …, B, G, R, B, G, R, B, G, R, B, …
  - …, B, R, G, B, R, G, B, R, G, B, …
- Blue nodes in Γ' make up directed Hamiltonian cycle Γ in G, or reverse of one. ▪

3-SAT Reduces to Directed Hamiltonian Cycle

Claim. 3-SAT $\leq_P$ DIR-HAM-CYCLE.

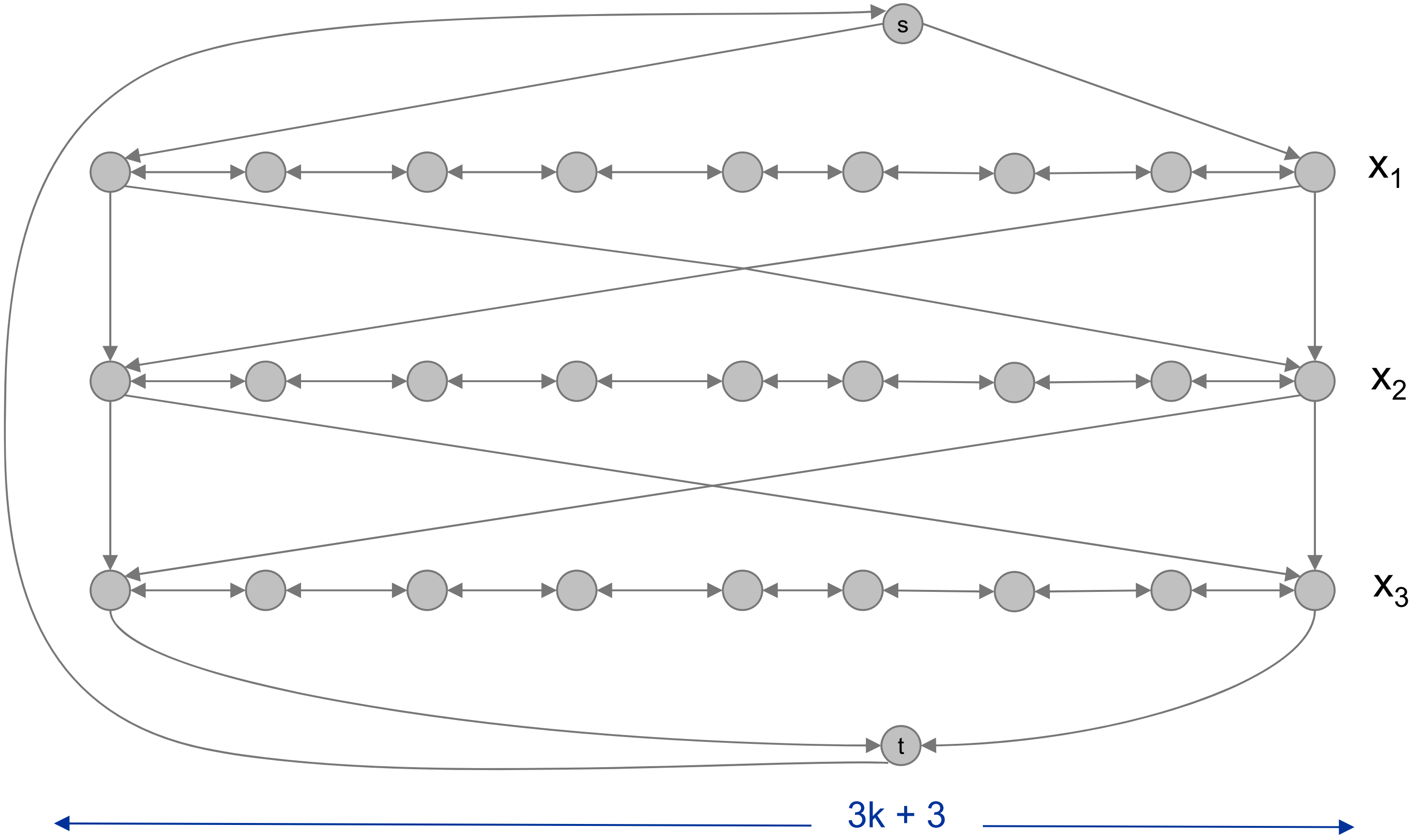
Pf. Given an instance Φ of 3-SAT, we construct an instance of DIR-HAM-CYCLE that has a Hamiltonian cycle iff Φ is satisfiable.

Construction. First, create graph that has 2^n Hamiltonian cycles which correspond in a natural way to 2^n possible truth assignments.

3-SAT Reduces to Directed Hamiltonian Cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

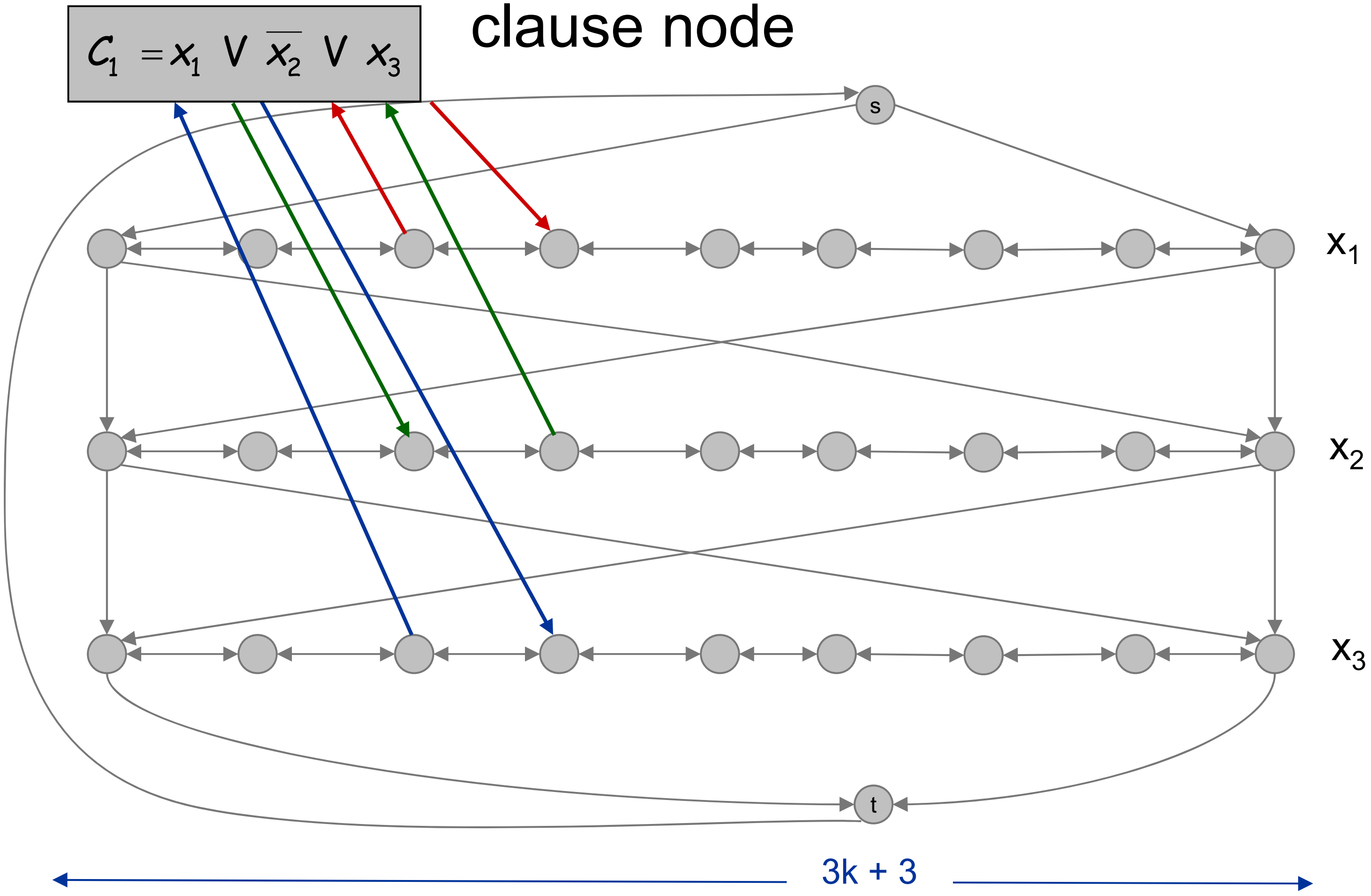
- Construct G to have 2^n Hamiltonian cycles.
- Intuition: traverse path i from left to right ⇔ set variable $x_i = 1$.



3-SAT Reduces to Directed Hamiltonian Cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

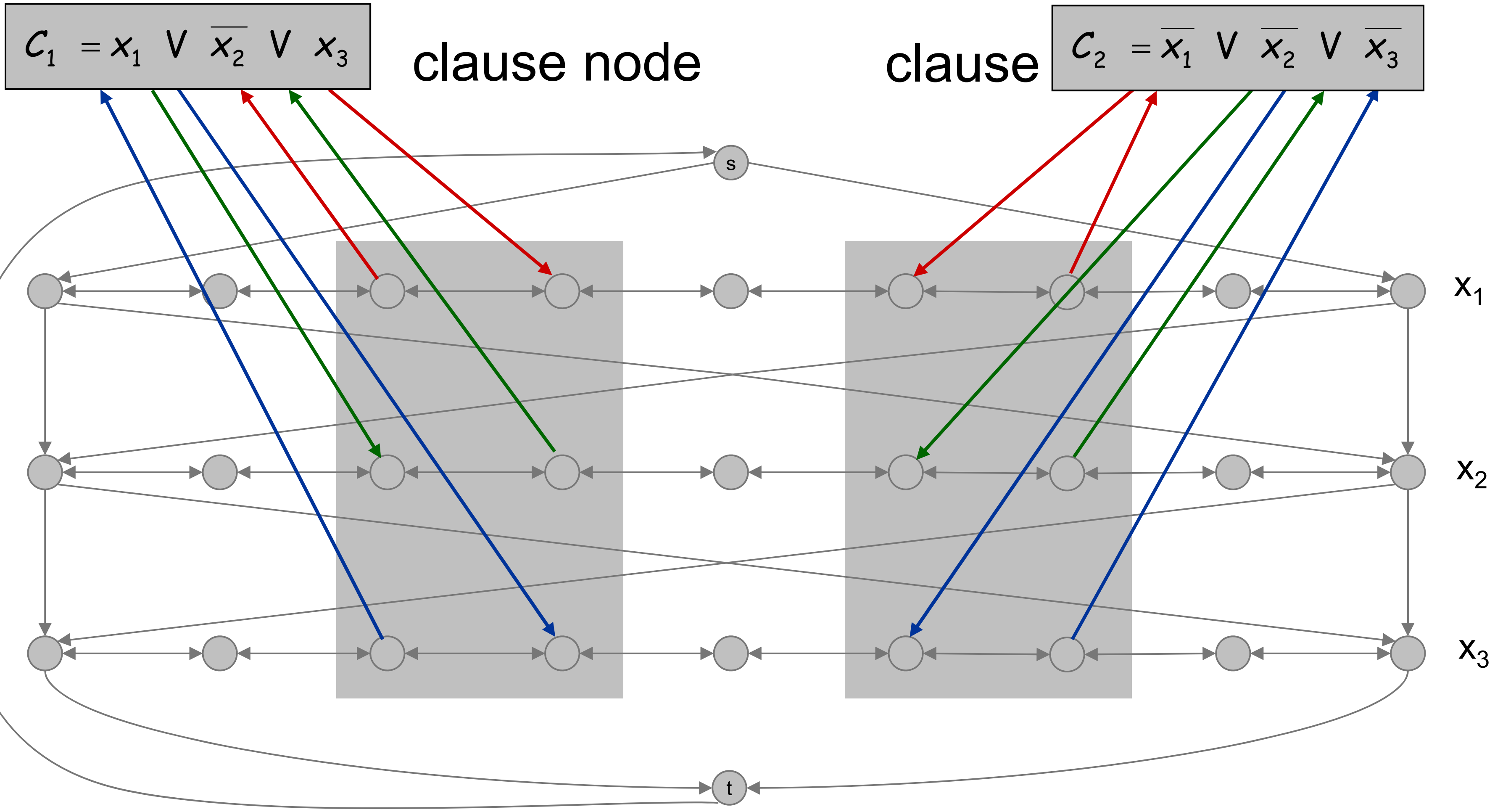
- Construct G to have 2^n Hamiltonian cycles.



3-SAT Reduces to Directed Hamiltonian Cycle

Construction. Given 3-SAT instance Φ with n variables x_i and k clauses.

- For each clause: add a node and 6 edges.



3-SAT Reduces to Directed Hamiltonian Cycle

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

Pf. $\Rightarrow$

- Suppose 3-SAT instance has satisfying assignment x^*.
- Then, define Hamiltonian cycle in G as follows:
  - if $x^*_i = 1$, traverse row i from left to right
  - if $x^*_i = 0$, traverse row i from right to left
  - for each clause C_j, there will be at least one row i in which we are going in "correct" direction to splice node C_j into tour

3-SAT Reduces to Directed Hamiltonian Cycle

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

Pf. $\Leftarrow$

- Suppose G has a Hamiltonian cycle Γ.
- If Γ enters clause node C_j , it must depart on mate edge.
  - thus, nodes immediately before and after C_j are connected by an edge e in G
  - removing C_j from cycle, and replacing it with edge e yields Hamiltonian cycle on G - { C_j }
- Continuing in this way, we are left with Hamiltonian cycle Γ' in
  G - { C_1 , C_2 , . . . , C_k }.
- Set x^*_i = 1 iff Γ' traverses row i left to right.
- Since Γ visits each clause node C_j , at least one of the paths is traversed in "correct" direction, and each clause is satisfied. ▪

Longest Path

SHORTEST-PATH. Given a digraph G = (V, E), does there exists a simple path of length at most k edges?

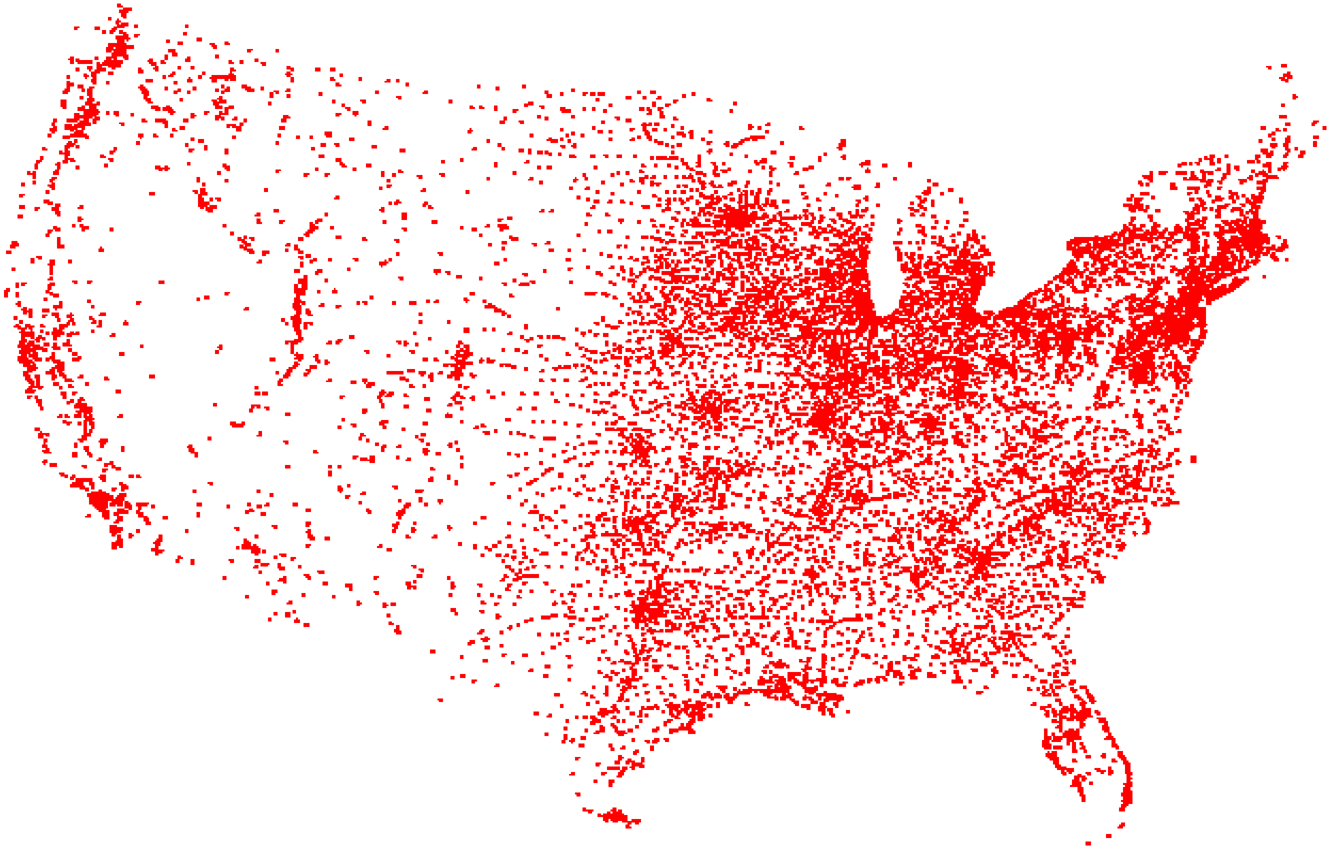
LONGEST-PATH. Given a digraph G = (V, E), does there exists a simple path of length at least k edges?

Claim. 3-SAT $\le_P$ LONGEST-PATH.

Pf 1. Redo proof for DIR-HAM-CYCLE, ignoring back-edge from t to s.
Pf 2. Show HAM-CYCLE $\le_P$ LONGEST-PATH.

Traveling Salesperson Problem

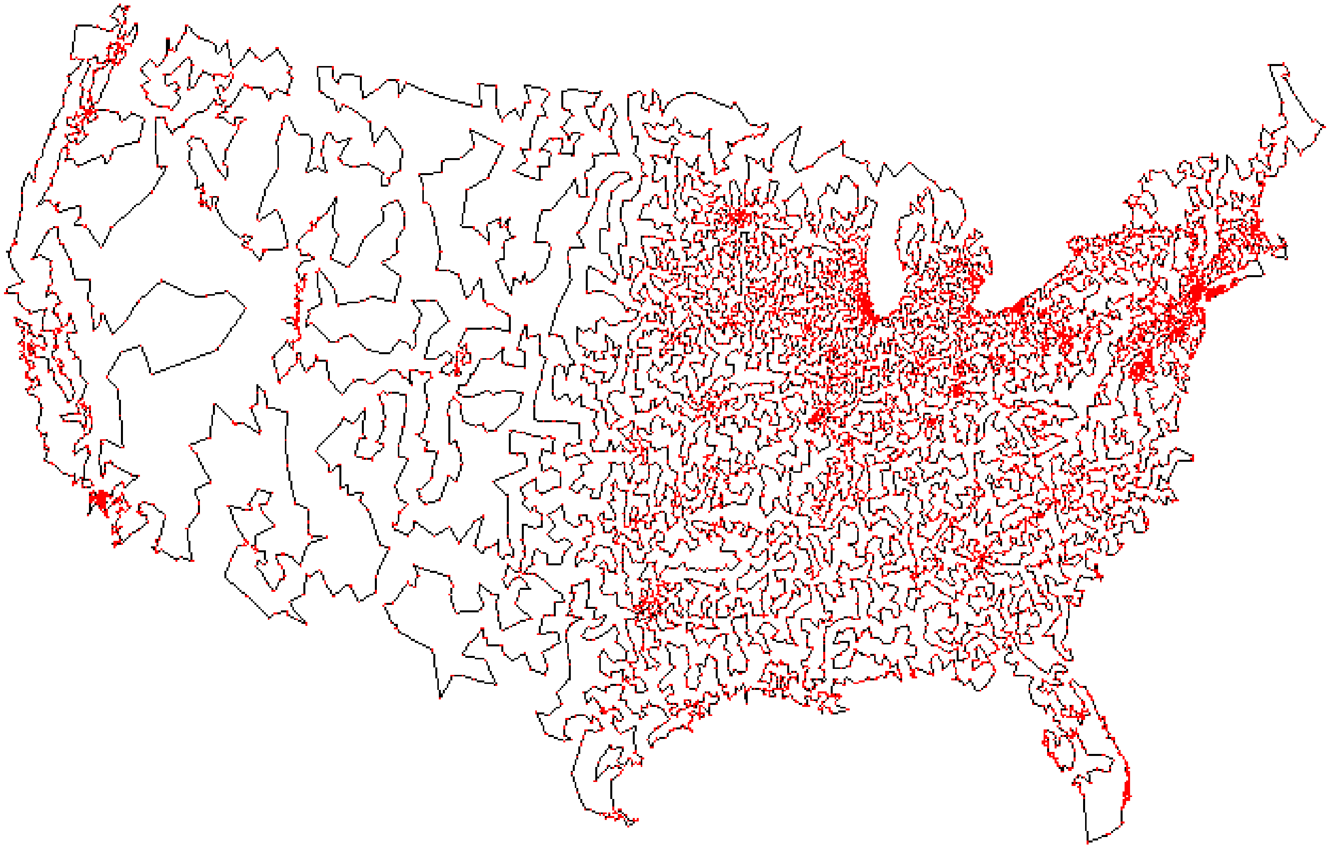
TSP. Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length ≤ D?



All 13,509 cities in US with a population of at least 500
Reference: http://www.tsp.gatech.edu

Traveling Salesperson Problem

TSP. Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length $\leq$ D?



Optimal TSP tour
Reference: http://www.tsp.gatech.edu

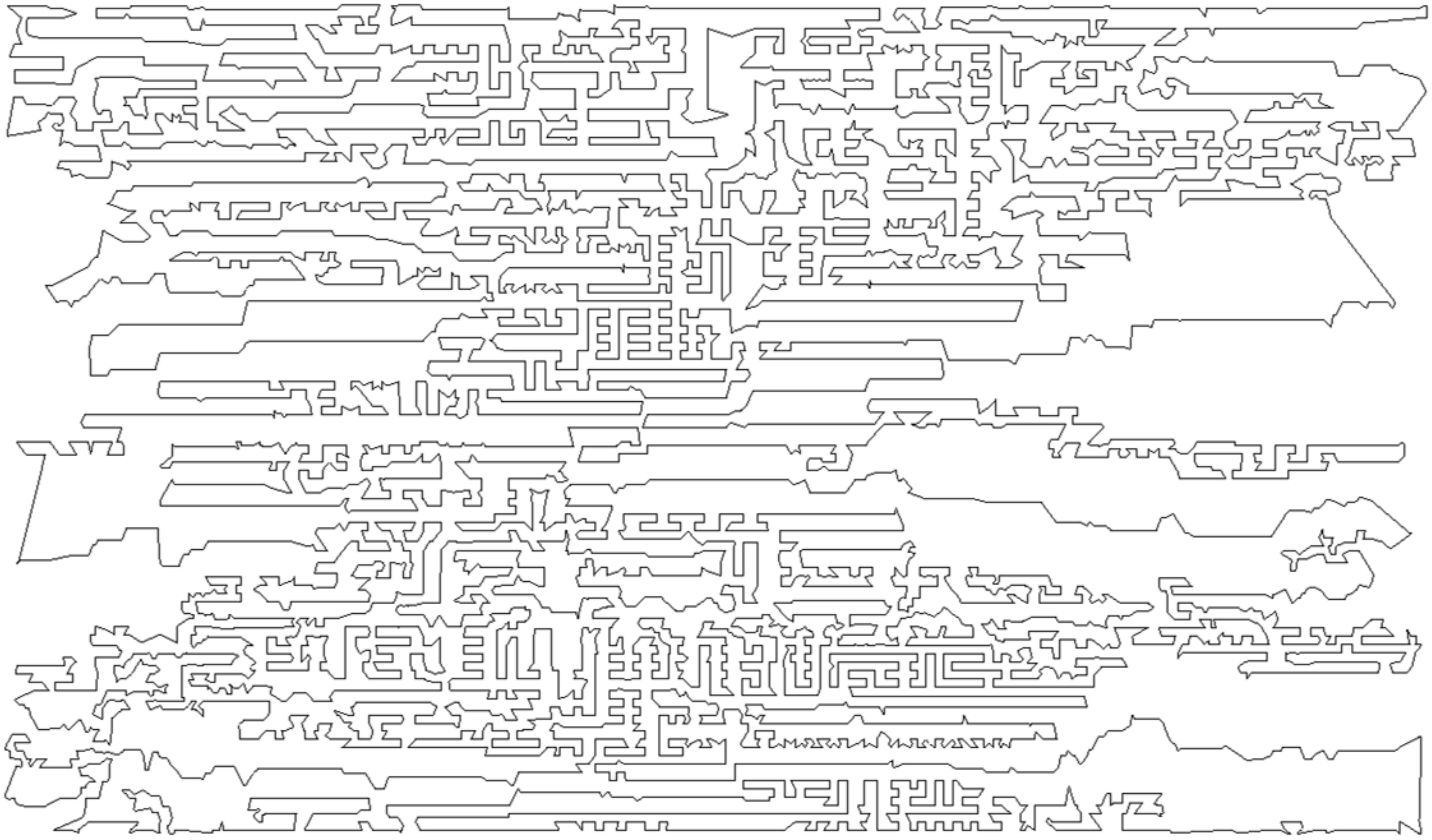
Traveling Salesperson Problem

TSP. Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length $\le$ D?

11,849 holes to drill in a programmed logic array
Reference: http://www.tsp.gatech.edu

Traveling Salesperson Problem

TSP. Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length ≤ D?



Optimal TSP tour
Reference: http://www.tsp.gatech.edu

Traveling Salesperson Problem

TSP. Given a set of n cities and a pairwise distance function d(u, v), is there a tour of length ≤ D?

HAM-CYCLE: given a graph G = (V, E), does there exists a simple cycle that contains every node in V?

Claim. HAM-CYCLE $\le_P$ TSP.

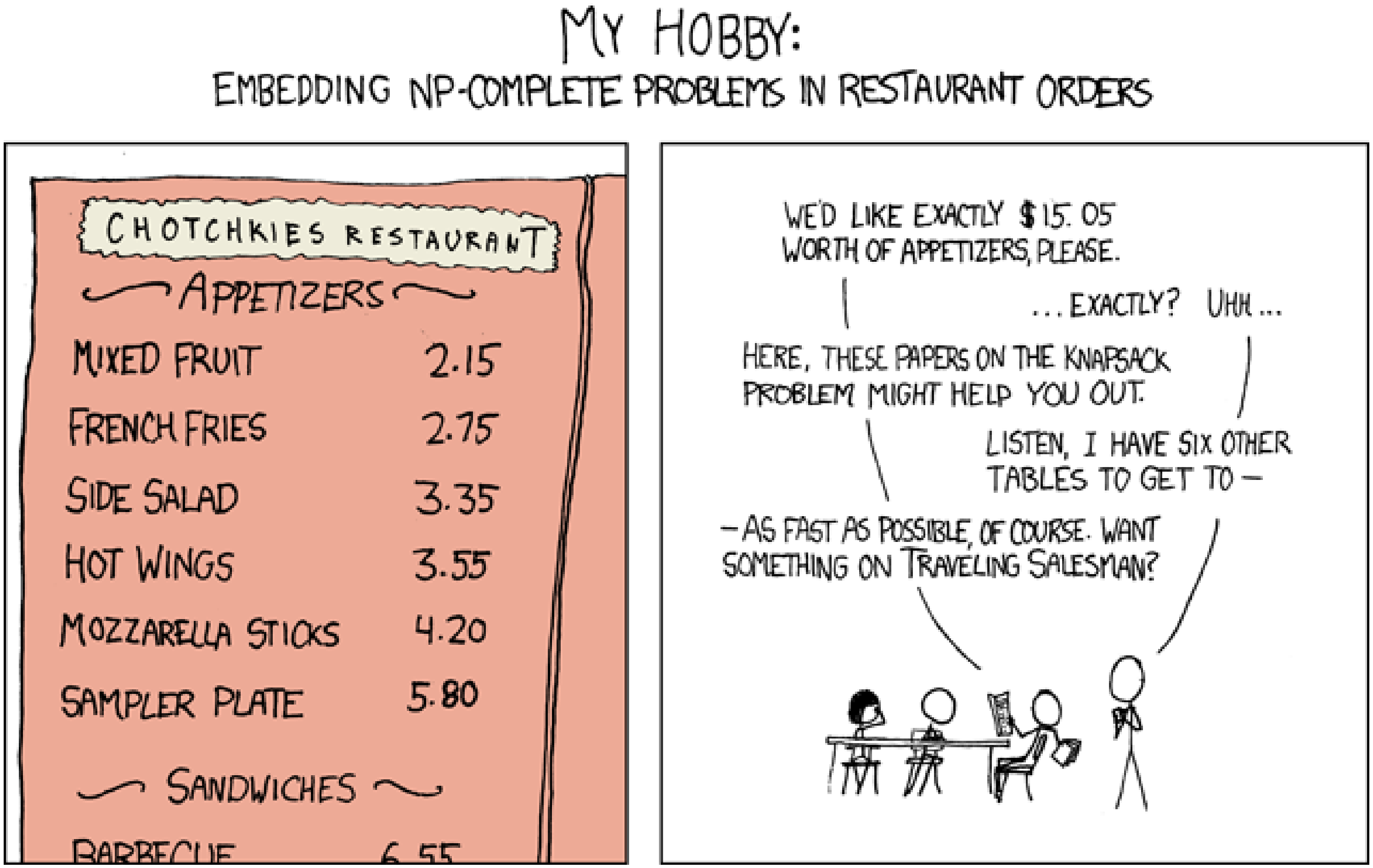
Pf.

- Given instance G = (V, E) of HAM-CYCLE, create n cities with distance function

$$d(u,\ v)\ =\ \begin{cases} 1 & \text{if } (u,\ v)\ \in\ E \\ 2 & \text{if } (u,\ v)\ \notin\ E \end{cases}$$

- TSP instance has tour of length ≤ n iff G is Hamiltonian. ▪

Remark. TSP instance in reduction satisfies Δ-inequality.



Randall Munro
`http://xkcd.com/c287.html`

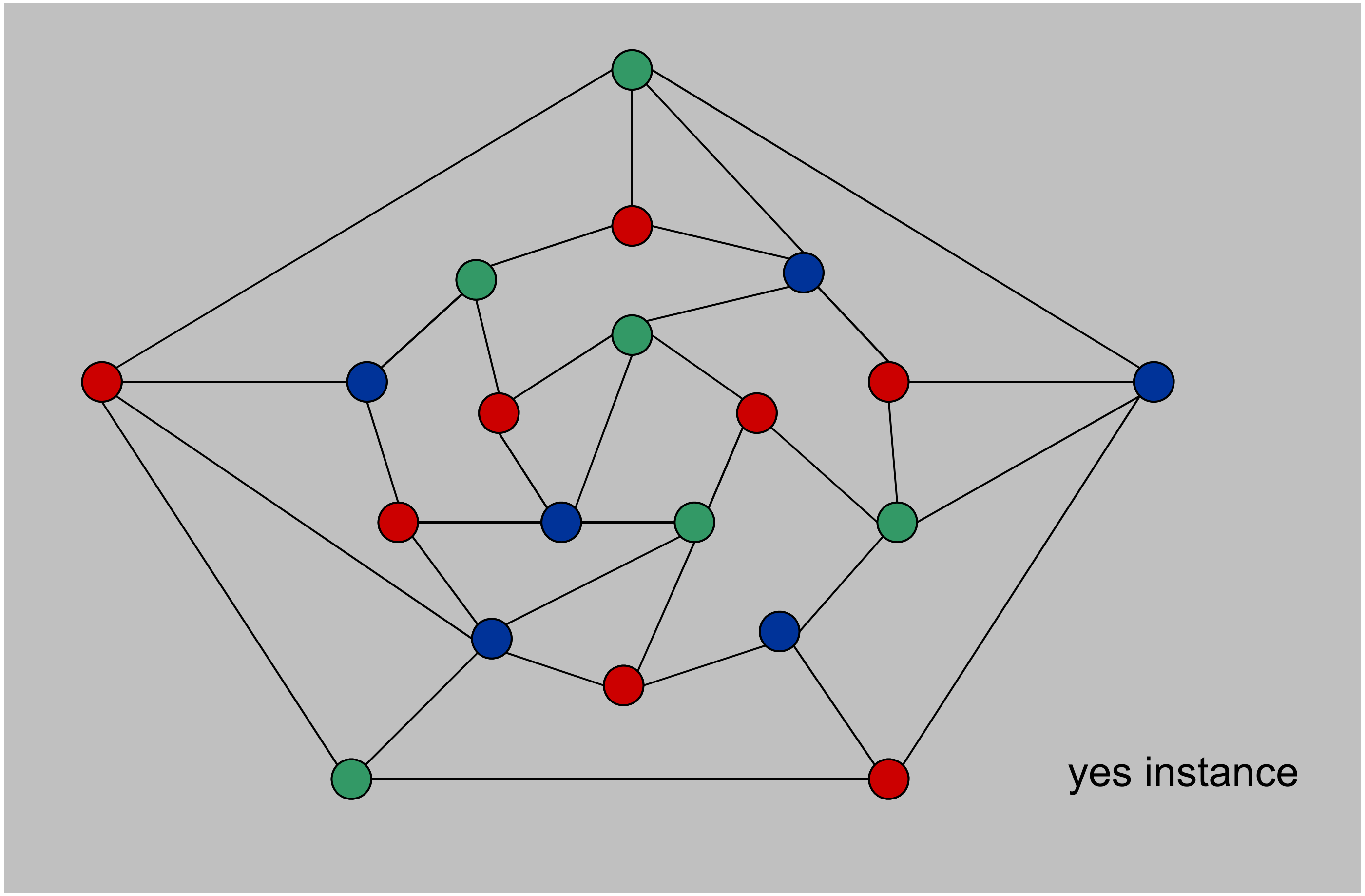
8.7 Graph Coloring

Basic genres.

- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING, 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.

3-Colorability

3-COLOR: Given an undirected graph G does there exists a way to color the nodes red, green, and blue so that no adjacent nodes have the same color?



Register Allocation

Register allocation. Assign program variables to machine register so that no more than k registers are used and no two program variables that are needed at the same time are assigned to the same register.

Interference graph. Nodes are program variables names, edge between u and v if there exists an operation where both u and v are "live" at the same time.

Observation. [Chaitin 1982] Can solve register allocation problem iff interference graph is k-colorable.

Fact. 3-COLOR $\le_P$ k-REGISTER-ALLOCATION for any constant $k \ge 3$.

3-Colorability

Claim. 3-SAT $\leq_P$ 3-COLOR.

Pf. Given 3-SAT instance Φ, we construct an instance of 3-COLOR that is 3-colorable iff Φ is satisfiable.

Construction.

i. For each literal, create a node.
ii. Create 3 new nodes T, F, B; connect them in a triangle, and connect each literal to B.
iii. Connect each literal to its negation.
iv. For each clause, add gadget of 6 nodes and 13 edges.

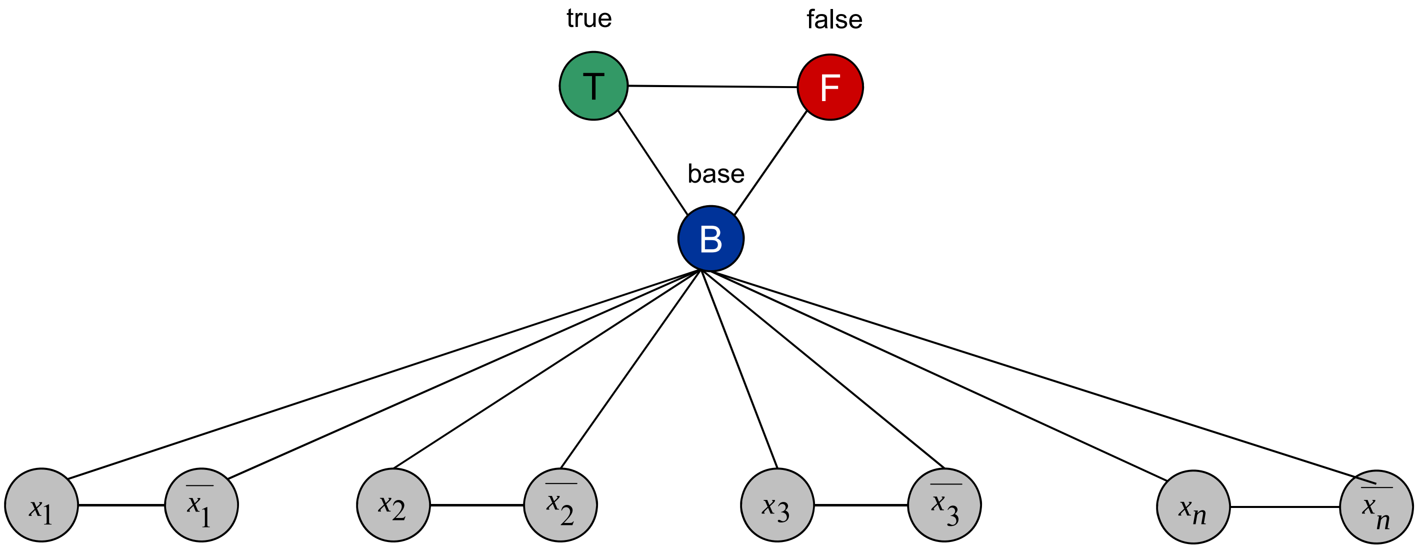
↑ to be described next

3-Colorability

Claim. Graph is 3-colorable iff Φ is satisfiable.

Pf. $\Rightarrow$ Suppose graph is 3-colorable.

- Consider assignment that sets all T literals to true.
- (ii) ensures each literal is T or F.
- (iii) ensures a literal and its negation are opposites.



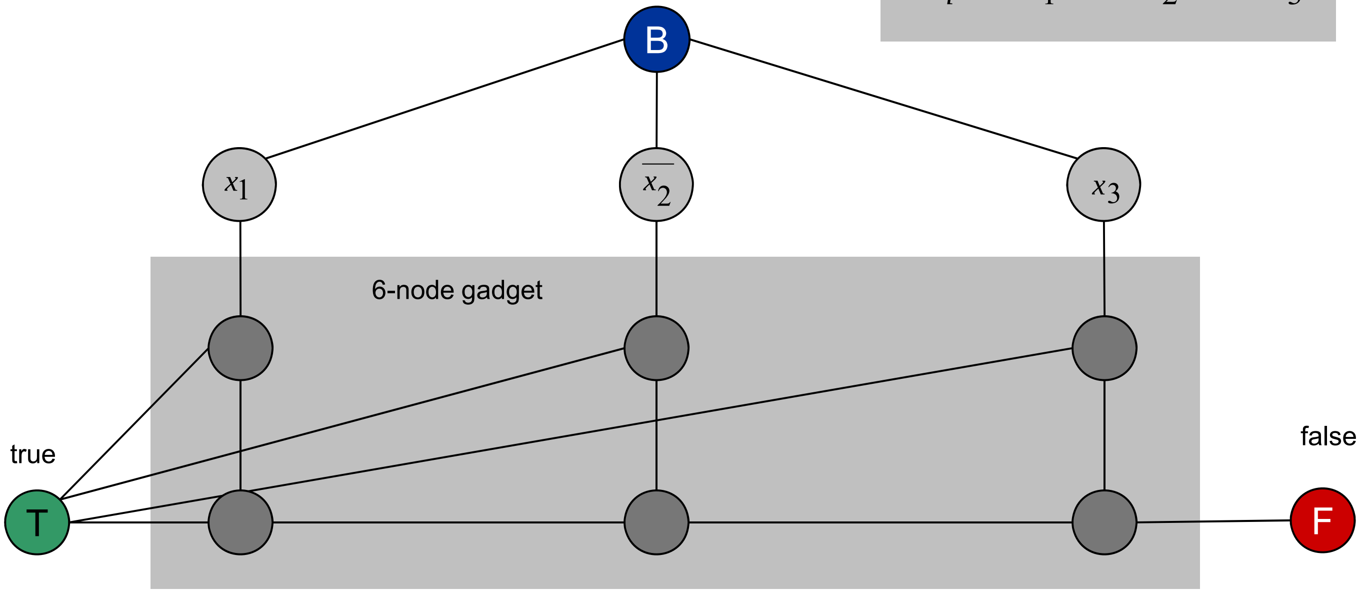
3-Colorability

Claim. Graph is 3-colorable iff Φ is satisfiable.

Pf. $\Rightarrow$ Suppose graph is 3-colorable.

- Consider assignment that sets all T literals to true.
- (ii) ensures each literal is T or F.
- (iii) ensures a literal and its negation are opposites.
- (iv) ensures at least one literal in each clause is T.

$$C_i = x_1 \text{ V } \overline{x_2} \text{ V } x_3$$



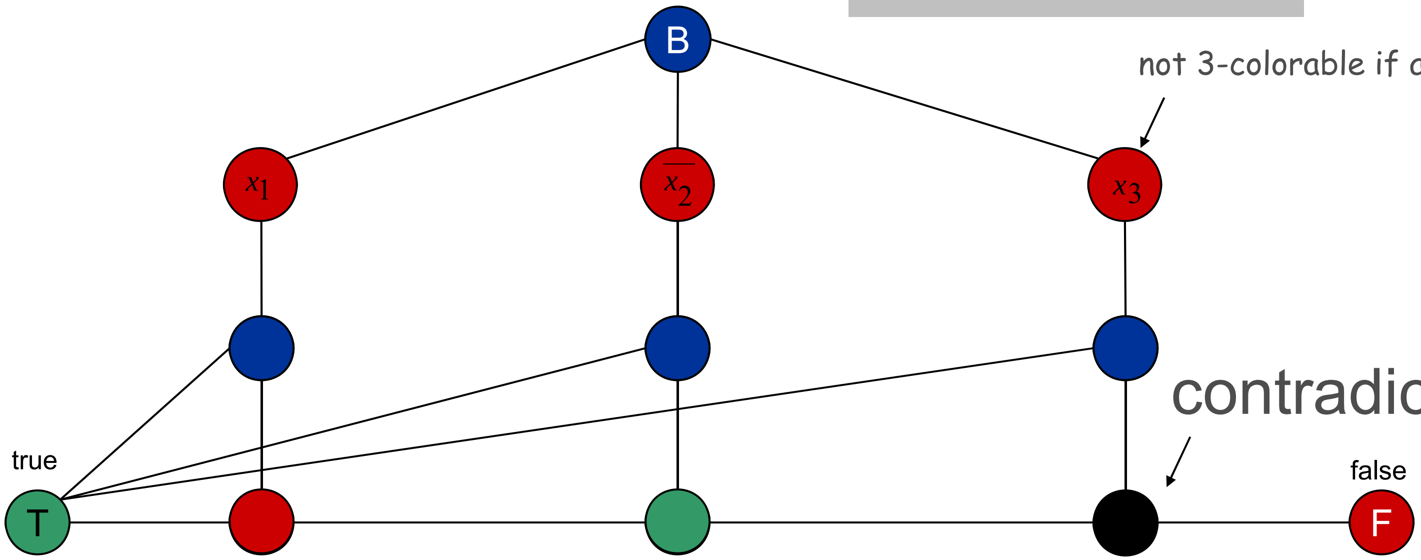
3-Colorability

Claim. Graph is 3-colorable iff Φ is satisfiable.

Pf. $\Rightarrow$ Suppose graph is 3-colorable.

- Consider assignment that sets all T literals to true.
- (ii) ensures each literal is T or F.
- (iii) ensures a literal and its negation are opposites.
- (iv) ensures at least one literal in each clause is T.

$$C_i = x_1 \text{ V } \overline{x_2} \text{ V } x_3$$



not 3-colorable if all are red

contradiction

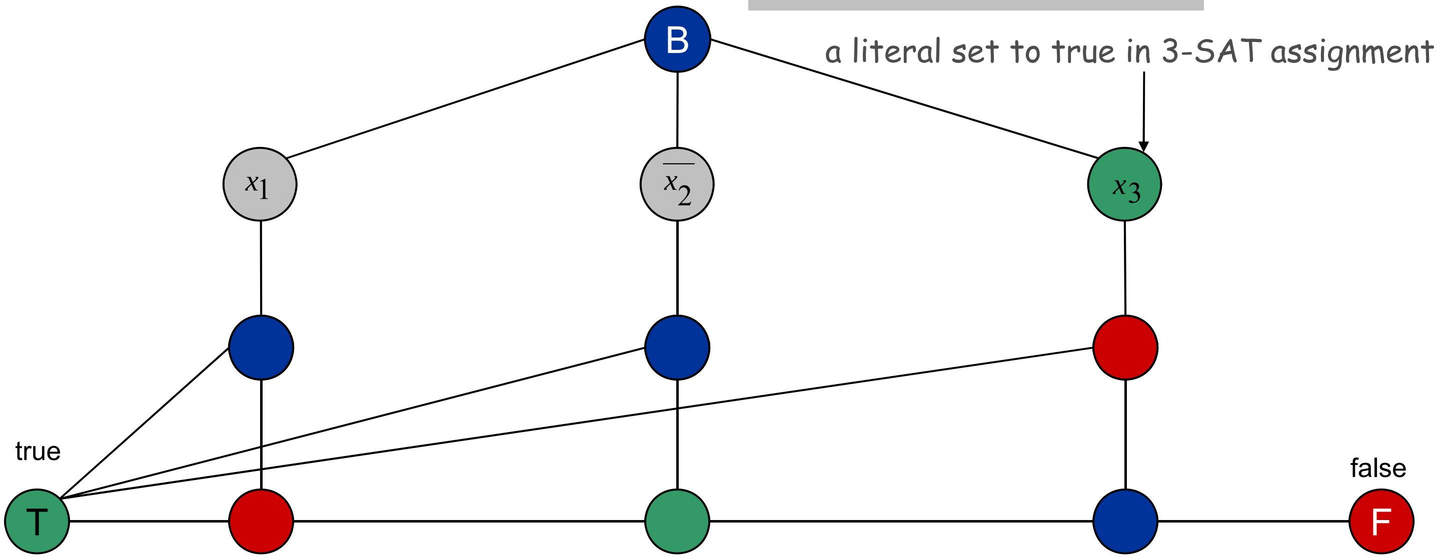
3-Colorability

Claim. Graph is 3-colorable iff Φ is satisfiable.

Pf. $\Leftarrow$ Suppose 3-SAT formula Φ is satisfiable.

- Color all true literals T.
- Color node below green node F, and node below that B.
- Color remaining middle row nodes B.
- Color remaining bottom nodes T or F as forced. ▪

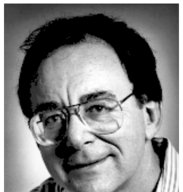
$$C_i = x_1 \ \text{V} \ \overline{x_2} \ \text{V} \ x_3$$



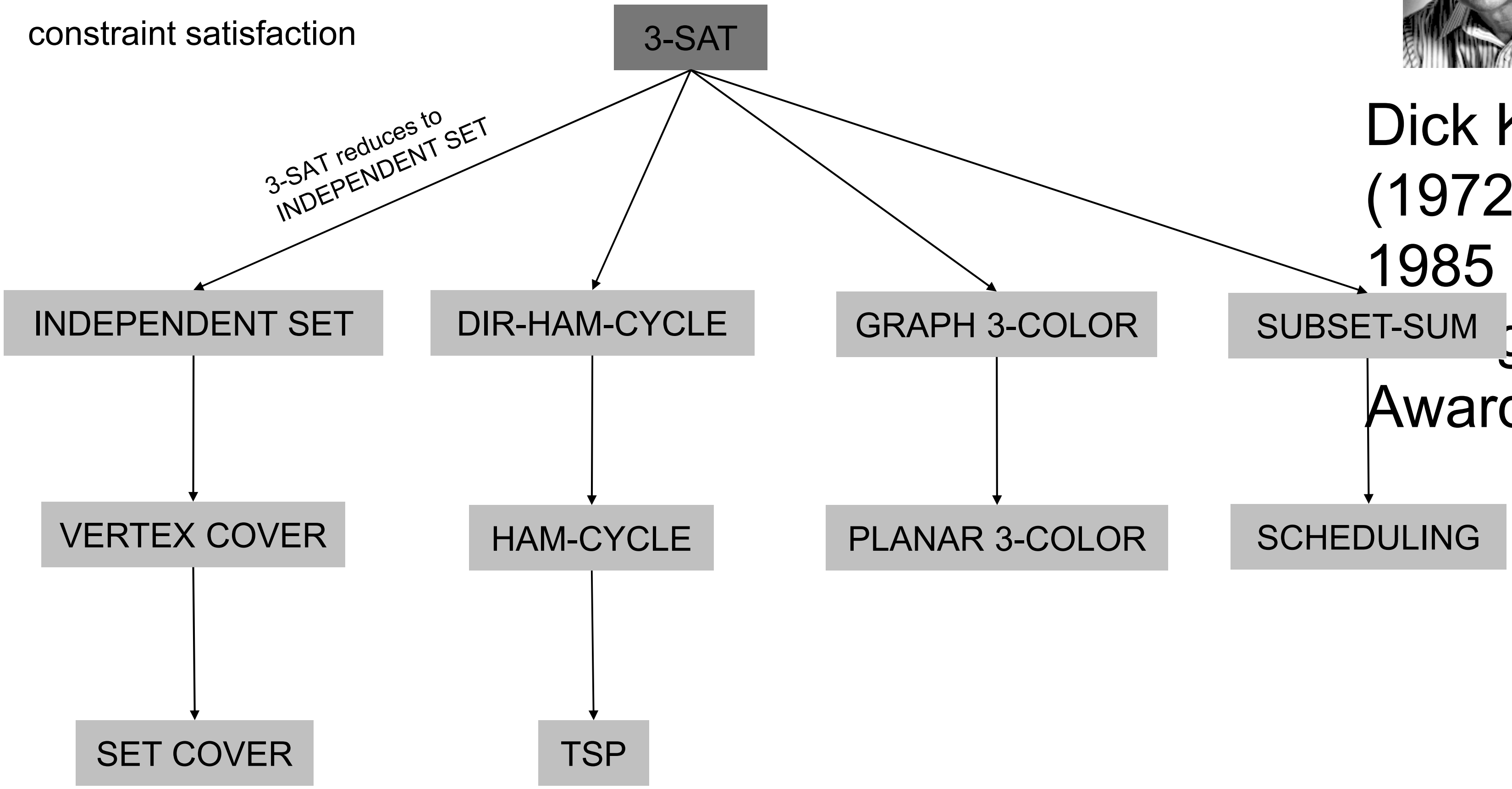
Extra Slides

8.10 A Partial Taxonomy of Hard Problems

Polynomial-Time Reductions



Dick Karp
(1972)
1985
g
Award



constraint satisfaction

packing and covering

sequencing

partitioning

numerical

Subset Sum (proof from book)

Construction. Let $X \cup Y \cup Z$ be a instance of 3D-MATCHING with triplet set T. Let $n = |X| = |Y| = |Z|$ and $m = |T|$.

- Let $X = \{ x_1, x_2, x_3 \ x_4 \}$, $Y = \{ y_1, y_2, y_3, y_4 \}$, $Z = \{ z_1, z_2, z_3, z_4 \}$
- For each triplet $t = (x_i, y_j, z_k) \in T$, create an integer w_t with 3n digits that has a 1 in positions i, $n+j$, and $2n+k$.

use base m+1

Claim. 3D-matching iff some subset sums to W = 111,…, 111.

Triplet t_i			x_1	x_2	x_3	x_4	y_1	y_2	y_3	y_4	z_1	z_2	z_3	z_4	w_i
x_1	y_2	z_3	1	0	0	0	0	1	0	0	0	0	1	0	**100,001,000,010**
x_2	y_4	z_2	0	1	0	0	0	0	0	1	0	1	0	0	**10,000,010,100**
x_1	y_1	z_1	1	0	0	0	1	0	0	0	1	0	0	0	**100,010,001,000**
x_2	y_2	z_4	0	1	0	0	0	1	0	0	0	0	0	1	**10,001,000,001**
x_4	y_3	z_4	0	0	0	1	0	0	1	0	0	0	0	1	**100,100,001**
x_3	y_1	z_2	0	0	1	0	1	0	0	0	0	1	0	0	**1,010,000,100**
x_3	y_1	z_3	0	0	1	0	1	0	0	0	0	0	1	0	**1,010,000,010**
x_3	y_1	z_1	0	0	1	0	1	0	0	0	1	0	0	0	**1,010,001,000**
x_4	y_4	z_4	0	0	0	1	0	0	0	1	0	0	0	1	**100,010,001**
															111,111,111,111

Partition

SUBSET-SUM. Given natural numbers $w_1, \ldots, w_n$ and an integer W, is there a subset that adds up to exactly W?

PARTITION. Given natural numbers $v_1, \ldots, v_m$, can they be partitioned into two subsets that add up to the same value?

$\nwarrow \frac{1}{2} \Sigma_i v_i$

Claim. SUBSET-SUM $\leq_P$ PARTITION.

Pf. Let $W, w_1, \ldots, w_n$ be an instance of SUBSET-SUM.

- Create instance of PARTITION with $m = n+2$ elements.
  - $v_1 = w_1, v_2 = w_2, \ldots, v_n = w_n, \quad v_{n+1} = 2 \Sigma_i w_i - W, \quad v_{n+2} = \Sigma_i w_i + W$
- There exists a subset that sums to W iff there exists a partition since two new elements cannot be in the same partition. ▪

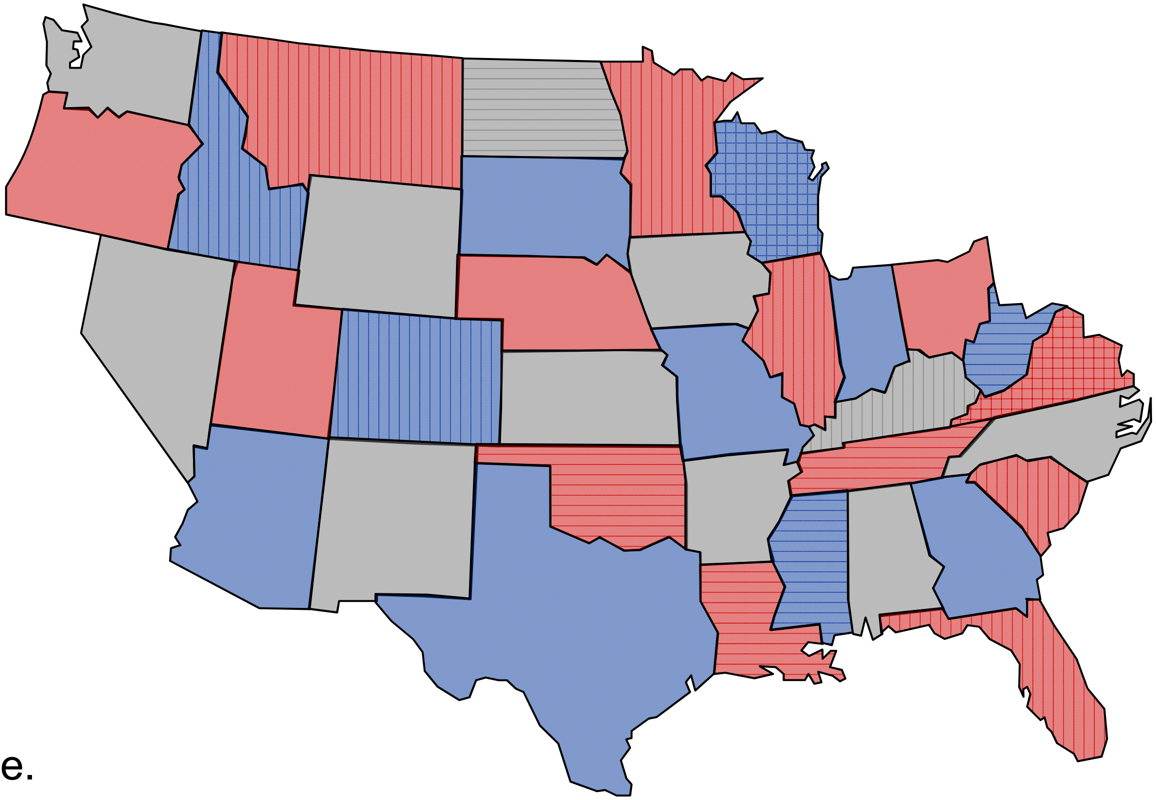
$v_{n+1} = 2 \Sigma_i w_i - W$ | W | subset A

$v_{n+2} = \Sigma_i w_i + W$ | $\Sigma_i w_i - W$ | subset B

4 Color Theorem

Planar 3-Colorability

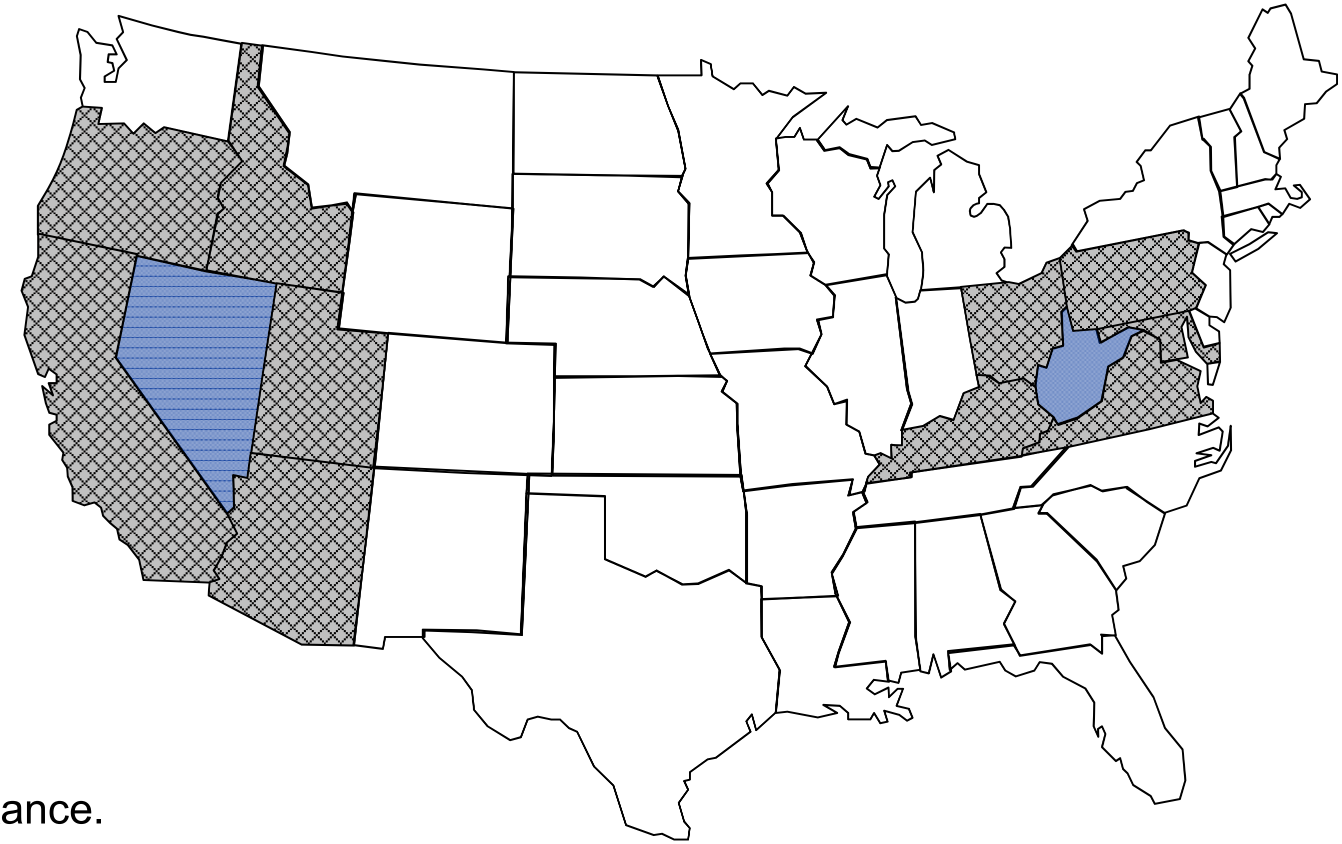
PLANAR-3-COLOR. Given a planar map, can it be colored using 3 colors so that no adjacent regions have the same color?



YES instance.

Planar 3-Colorability

PLANAR-3-COLOR. Given a planar map, can it be colored using 3 colors so that no adjacent regions have the same color?

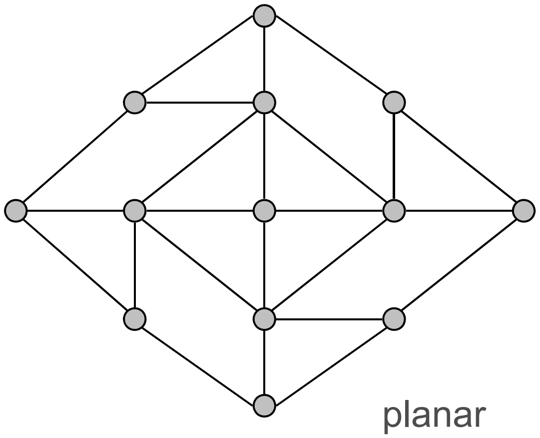


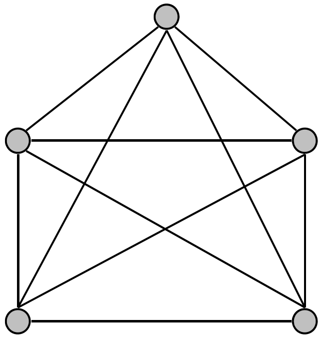
NO instance.

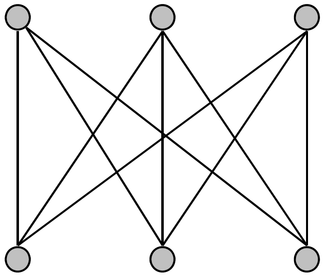
Planarity

Def. A graph is planar if it can be embedded in the plane in such a way that no two edges cross.

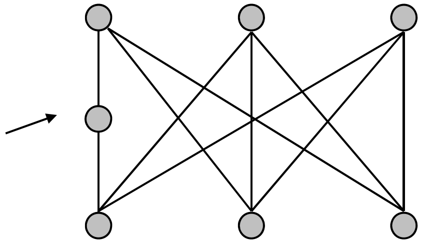
Applications: VLSI circuit design, computer graphics.


planar


K_5: non-planar


$K_{3,3}$: non-planar

Kuratowski's Theorem. An undirected graph G is non-planar iff it contains a subgraph homeomorphic to K_5 or $K_{3,3}$.


homeomorphic to $K_{3,3}$

Planarity Testing

Planarity testing. [Hopcroft-Tarjan 1974] O(n).

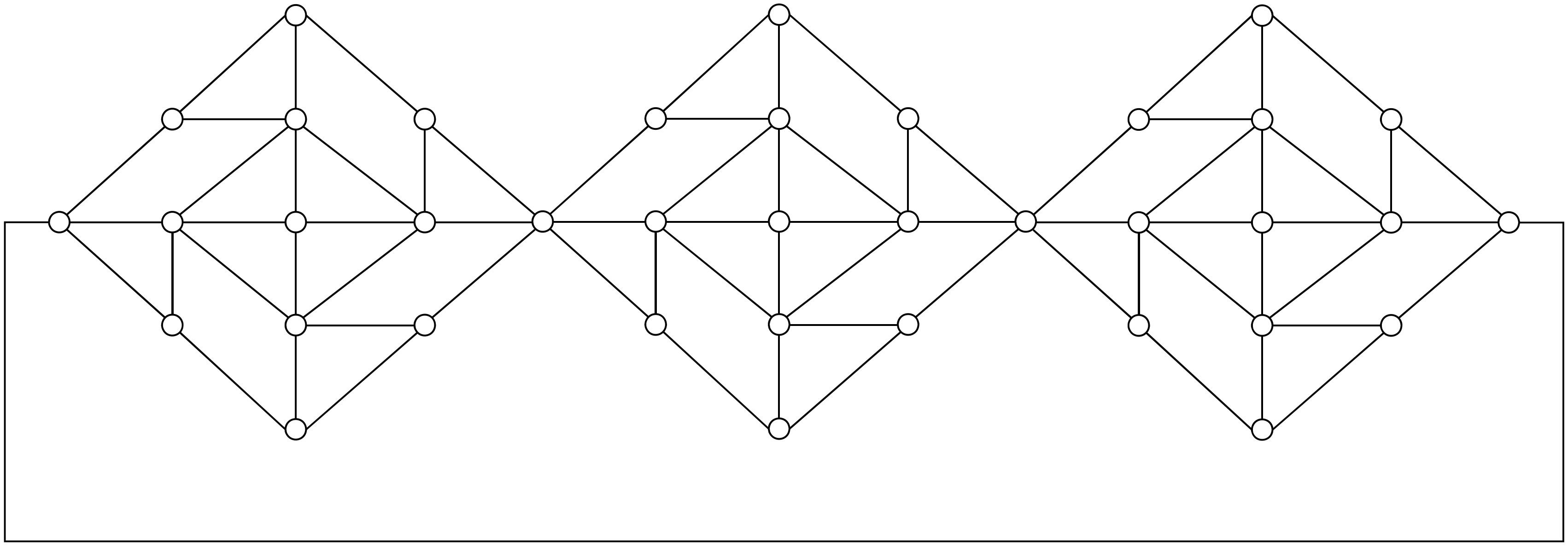
↑

simple planar graph can have at mos

Remark. Many intractable graph problems can be solved in poly-time if the graph is planar; many tractable graph problems can be solved faster if the graph is planar.

Planar Graph 3-Colorability

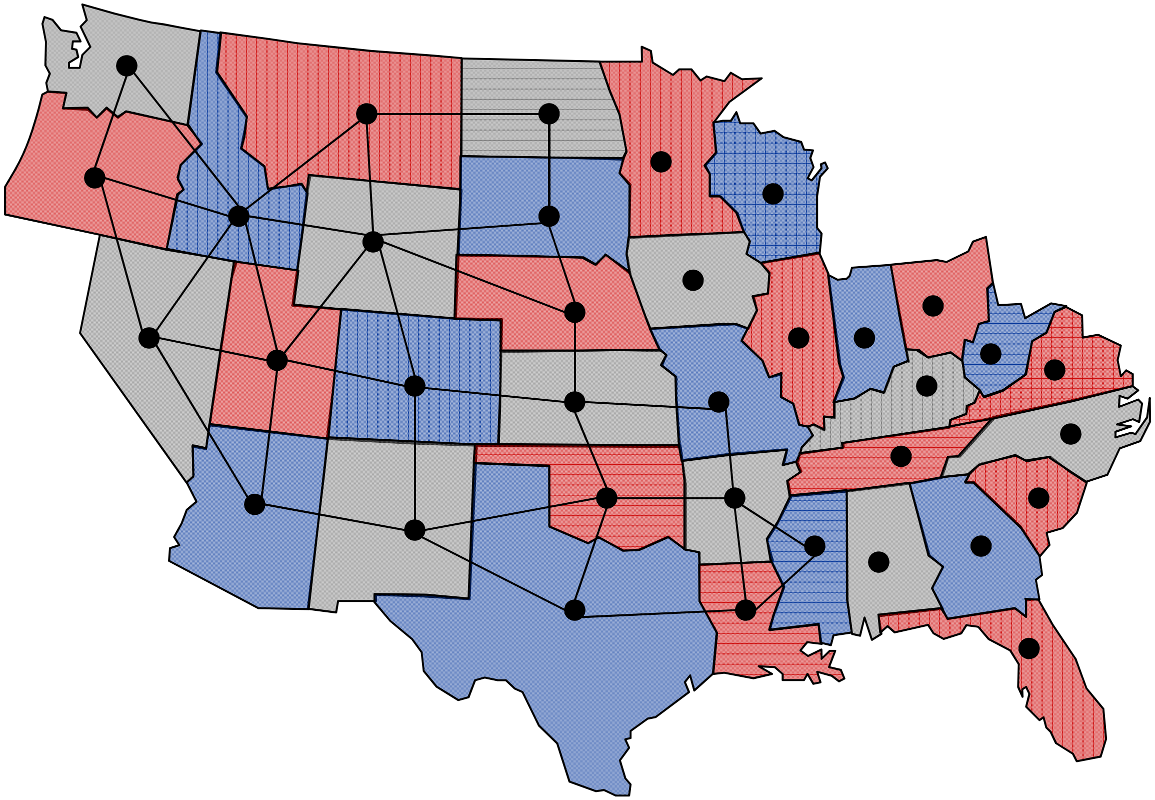
Q. Is this planar graph 3-colorable?



Planar 3-Colorability and Graph 3-Colorability

Claim. PLANAR-3-COLOR $\leq_P$ PLANAR-GRAPH-3-COLOR.

Pf sketch. Create a vertex for each region, and an edge between regions that share a nontrivial border.

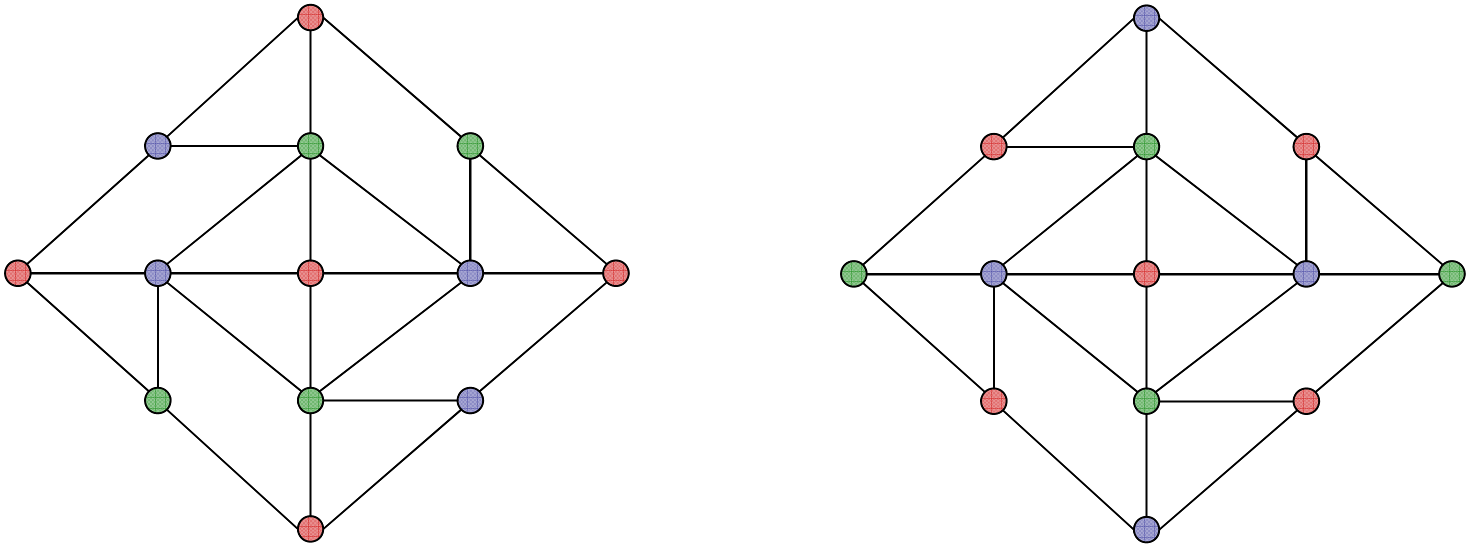


Planar Graph 3-Colorability

Claim. W is a planar graph such that:

- In any 3-coloring of W, opposite corners have the same color.
- Any assignment of colors to the corners in which opposite corners have the same color extends to a 3-coloring of W.

Pf. Only 3-colorings of W are shown below (or by permuting colors).

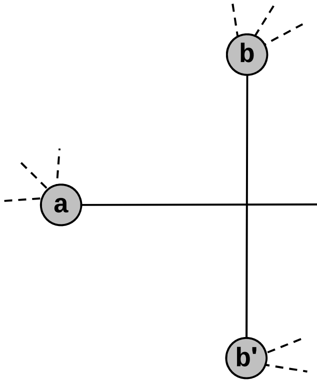


Planar Graph 3-Colorability

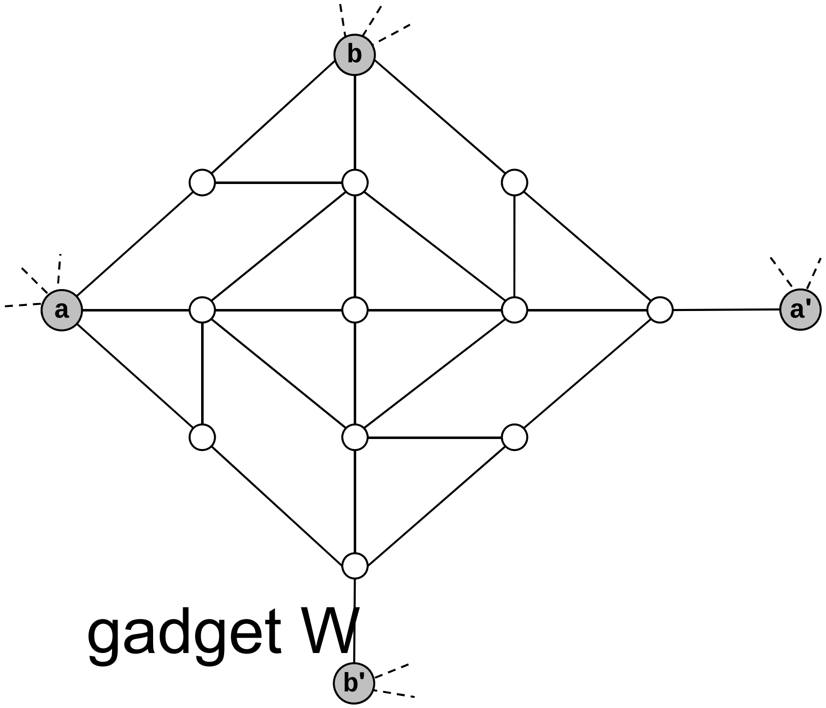
Claim. 3-COLOR $\leq_P$ PLANAR-GRAPH-3-COLOR.

Pf. Given instance of 3-COLOR, draw graph in plane, letting edges cross.

- Replace each edge crossing with planar gadget W.
- In any 3-coloring of W, $a \neq a'$ and $b \neq b'$.
- If $a \neq a'$ and $b \neq b'$ then can extend to a 3-coloring of W.



a crossing



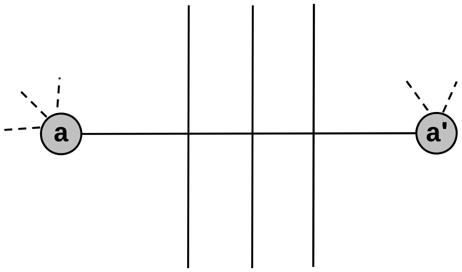
gadget W

Planar Graph 3-Colorability

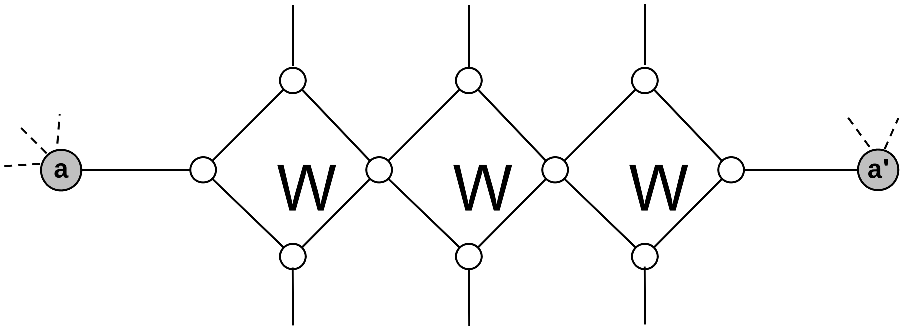
Claim. 3-COLOR $\leq_P$ PLANAR-GRAPH-3-COLOR.

Pf. Given instance of 3-COLOR, draw graph in plane, letting edges cross.

- Replace each edge crossing with planar gadget W.
- In any 3-coloring of W, $a \neq a'$ and $b \neq b'$.
- If $a \neq a'$ and $b \neq b'$ then can extend to a 3-coloring of W.

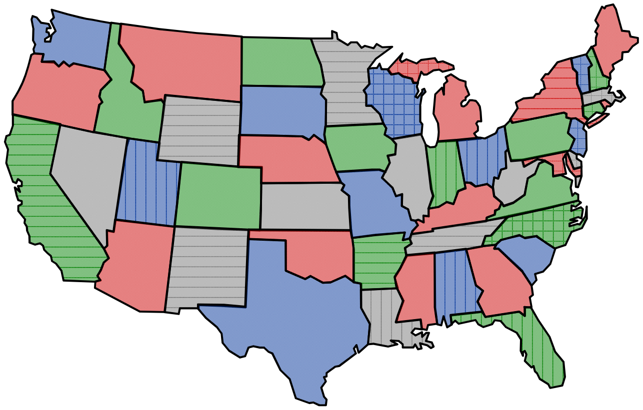


multiple crossings



gadget W

Planar k-Colorability



PLANAR-2-COLOR. Solvable in linear time.

PLANAR-3-COLOR. NP-complete.

PLANAR-4-COLOR. Solvable in O(1) time.

Theorem. [Appel-Haken, 1976] Every planar map is 4-colorable.

- Resolved century-old open problem.
- Used 50 days of computer time to deal with many special cases.
- First major theorem to be proved using computer.

False intuition. If PLANAR-3-COLOR is hard, then so is PLANAR-4-COLOR and PLANAR-5-COLOR.

Polynomial-Time Detour

Graph minor theorem. [Robertson-Seymour 1980s]

Corollary. There exist an O(n^3) algorithm to determine if a graph can be embedded in the torus in such a way that no two edges cross.

Pf of theorem. Tour de force.

Polynomial-Time Detour

Graph minor theorem. [Robertson-Seymour 1980s]

Corollary. There exist an O(n^3) algorithm to determine if a graph can be embedded in the torus in such a way that no two edges cross.

Mind boggling fact 1. The proof is highly non-constructive!
Mind boggling fact 2. The constant of proportionality is enormous!

> Unfortunately, for any instance G = (V, E) that one could fit into the known universe, one would easily prefer n^{70} to even *constant* time, if that constant had to be one of Robertson and Seymour's. - David Johnson

Theorem. There exists an explicit O(n) algorithm.
Practice. LEDA implementation guarantees O(n^3).