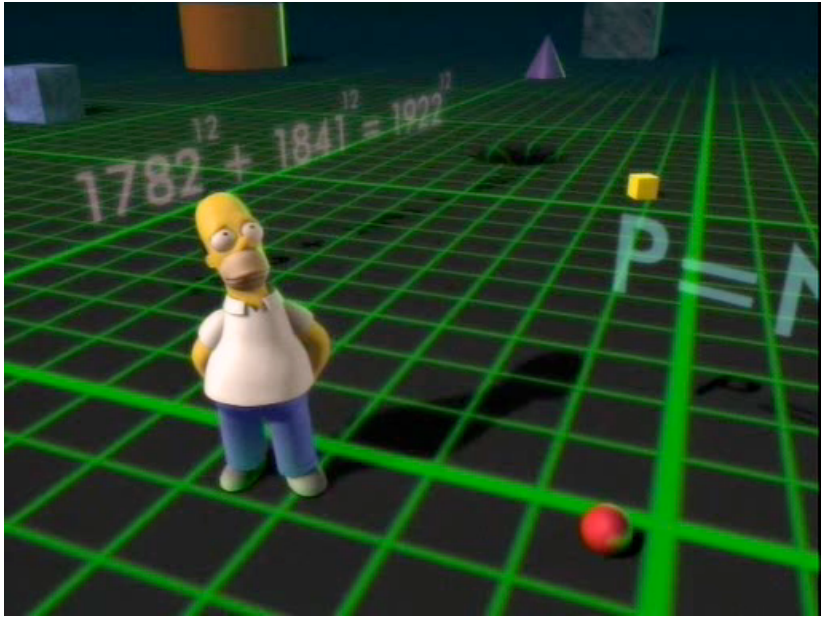
chang@umbc.edu

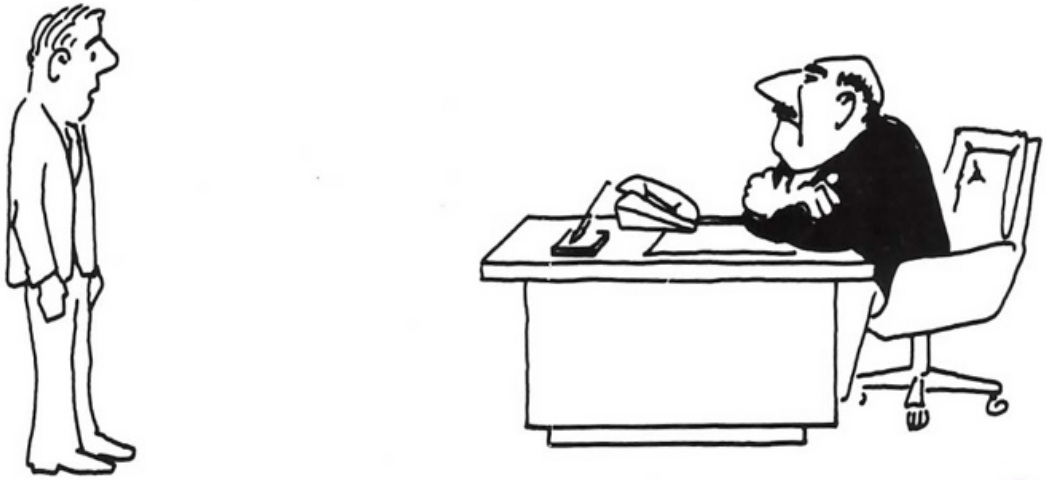
NP Completeness: Reductions

not "priced for quick sale"


1782^12 + 1841^12 = 1922^12
P=NP


P=NP
e^πi = -1


P=NP



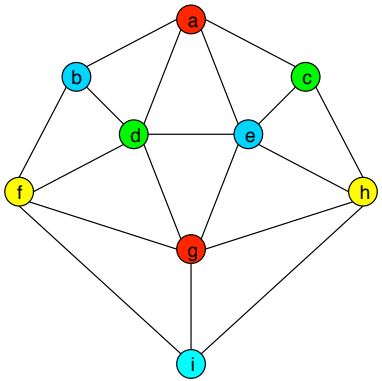
"I can't find an efficient algorithm, I guess I'm just too dumb."

[Garey & Johnson, '79]

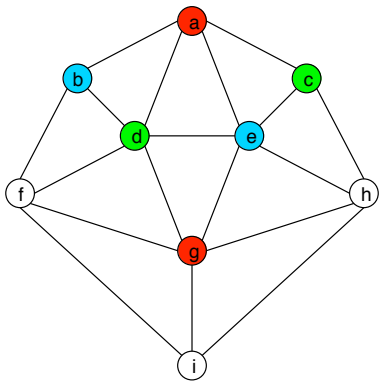
"I can't find an efficient algorithm, because no such algorithm is possible!"

"I can't find an efficient algorithm, but neither can all these famous people."

Graph Coloring: assign colors to vertices so that adjacent vertices have different colors



4-colorable



not 3-colorable

f cannot be red, green or blue

yes: $\wedge \vee \neg ( )$ no: $\forall \exists$

Satisfiability: given a Boolean formula, does there exist an assignment of truth values to the variables that makes the formula true?

↑ i.e. does any row of the truth table evaluate to true?

$$\varphi = (x_1 \vee \overline{x_3} \vee \overline{x_4}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_4}) \wedge (x_2 \vee \overline{x_3} \vee x_4)$$

↖ conjunctive normal form = product-of-sums

satisfiable: $x_1 = 1$, $x_2 = 0$, $x_3 = 0$, $x_4 = 0$

$$\varphi = (x_1 \vee x_2) \wedge (\overline{x_1} \vee x_2) \wedge (x_1 \vee \overline{x_2}) \wedge (\overline{x_1} \vee \overline{x_2})$$

not satisfiable.

Note: truth table have an exponential # of rows.

Decision problems:

$3COLOR = \{ G \mid G$ is an undirected graph that is 3-colorable $\}$

$SAT = \{ \varphi \mid \varphi$ is a Boolean formula that is satisfiable $\}$

They look very different but are very much related

① given G, we can construct φ' s.t.

$$G \in 3COLOR \iff \varphi' \in SAT$$

easier if φ is in 3CNF

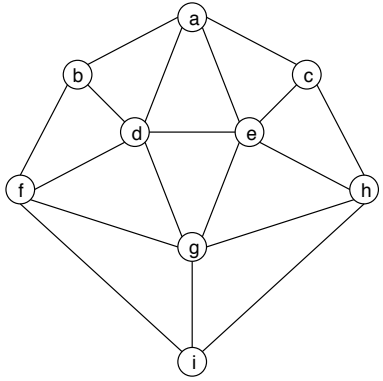
② given φ, we can construct G' s.t.

$$\varphi \in SAT \iff G' \in 3COLOR$$

If you have a fast algorithm for one then you have a fast algorithm for the other.

$G \mapsto \varphi'$ example

G has 9 vertices & 18 edges $\longmapsto$ φ' has 27 variables & 90 clauses



colors: Red, Green, Blue

3 variables per vertex

x_{uR} means vertex u is Red
x_{uG} means vertex u is Green
x_{uB} means vertex u is Blue

} intent only. Not part of the construction.

all 27 variables

x_{aR}	x_{bR}	x_{cR}	x_{dR}	x_{eR}	x_{fR}	x_{gR}	x_{hR}	x_{iR}
x_{aG}	x_{bG}	x_{cG}	x_{dG}	x_{eG}	x_{fG}	x_{gG}	x_{hG}	x_{iG}
x_{aB}	x_{bB}	x_{cB}	x_{dB}	x_{eB}	x_{fB}	x_{gB}	x_{hB}	x_{iB}

$\varphi' = \varphi'_1 \wedge \varphi'_2 \wedge \varphi'_3$

φ'_1: each vertex has at least 1 color

φ'_2: no vertex has 2 colors

φ'_3: adjacent vertices have different colors

$$\varphi_1' = (x_{aR} \vee x_{aG} \vee x_{aB}) \wedge (x_{bR} \vee x_{bG} \vee x_{bB}) \wedge \vdots \wedge (x_{iR} \vee x_{iG} \vee x_{iB})$$

9 clauses

vertex a is Red or Green or Blue
vertex b is Red or Green or Blue
⋮

Suppose that some assignment of 0-1 values to the variables makes φ_1' evaluate to 1.

0 = false
1 = true

Then in that assignment, every vertex has at least one color.

we want exactly 1

$$\varphi_2' = (\overline{x_{aR}} \vee \overline{x_{aG}}) \wedge (\overline{x_{aR}} \vee \overline{x_{aB}}) \wedge (\overline{x_{aG}} \vee \overline{x_{aB}}) \wedge (\overline{x_{bR}} \vee \overline{x_{bG}}) \wedge \cdots \wedge (\overline{x_{iG}} \vee \overline{x_{iB}})$$

27 clauses

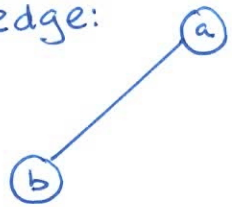
a not Red and Green at the same time
a not Red and Blue at the same time
a not Green and Blue at the same time
⋮

$(\overline{y} \vee \overline{z})$: if both $y=1$ and $z=1$, then $\overline{y} \vee \overline{z} = 0$

If φ_2' evaluates to 1, then assignment gives each vertex < 2 colors.

$\varphi_1' \wedge \varphi_2'$ means each vertex has exactly 1 color. ← for a satisfying assignment.

edge:



We want to make sure that vertex a & vertex b have different colors.

a is Red $\Rightarrow$ b is not Red

$(\overline{x_{aR}} \vee \overline{x_{bR}})$

a is Green $\Rightarrow$ b is not Green

a is Blue $\Rightarrow$ b is not Blue

$$\varphi_3' = (\overline{x_{aR}} \vee \overline{x_{bR}}) \wedge (\overline{x_{aG}} \vee \overline{x_{bG}}) \wedge (\overline{x_{aB}} \vee \overline{x_{bB}})$$ ← a & b different colors in a satisfying assgn.

$\wedge$ ⋮

18 x 3 = 54 clauses

$$\wedge (\overline{x_{hR}} \vee \overline{x_{iR}}) \wedge (\overline{x_{hG}} \vee \overline{x_{iG}}) \wedge (\overline{x_{hB}} \vee \overline{x_{iB}})$$ h & i different colors

Recap: $\varphi' = \varphi_1' \wedge \varphi_2' \wedge \varphi_3'$ 27 variables

$(9 + 27 + 54) = 90$ clauses

Need to argue $G \in 3COLOR \iff \varphi' \in SAT$

($\Rightarrow$) If G is 3-colorable, then every vertex of G is assigned one of {Red, Green, Blue}

For each vertex u, if u is assigned Red, $x_{uR}=1$, $x_{uG}=0$, $x_{uB}=0$

if u is assigned Green, $x_{uR}=0$, $x_{uG}=1$, $x_{uB}=0$

if u is assigned Blue, $x_{uR}=0$, $x_{uG}=0$, $x_{uB}=1$.

φ_1' and φ_2' are satisfied since each vertex has exactly one color.

φ_3' is satisfied because we started w/ a legal coloring of G.
No two adjacent vertices are assigned the same color.

need this direction. Otherwise, we can just output φ'_1 that is always true.

($\Leftarrow$)

For all vertices u, exactly one of x_{uR}, x_{uG} & x_{uB} is assigned 1, since φ'_1 and φ'_2 are both satisfied. Assign vertex u the corresponding color. Since φ'_3 is also satisfied, no two adjacent vertices are assigned same color.

$\boxtimes$

G is 3-colorable $\Rightarrow$ φ' is satisfiable

G is not 3-colorable $\Rightarrow$ φ' is not satisfiable.

If we have an algorithm for SAT, then we can use it to determine if a graph is 3-colorable.

Recap: Efficient transformation of an instance of 3-COLOR into an instance of SAT.

Defn: we say that 3COLOR reduces to SAT.

Idea: if there's an efficient algorithm A for SAT, then there's an efficient algorithm for 3COLOR.

1. $G \mapsto \varphi'$
2. Run A on φ'
3. if A says $\varphi' \in$ SAT, output YES
4. o.w. Output NO

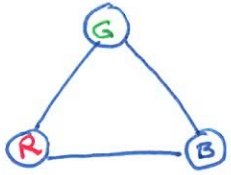
What if SAT is much harder than 3-COLOR?

$\varphi \mapsto G'$ (other direction)

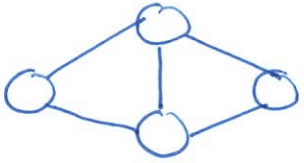
Given a Boolean formula φ in 3CNF, construct an undirected graph G' s.t.

$$\varphi \in \text{SAT} \iff G' \in \text{3COLOR}$$

Some "gadgets":



palette

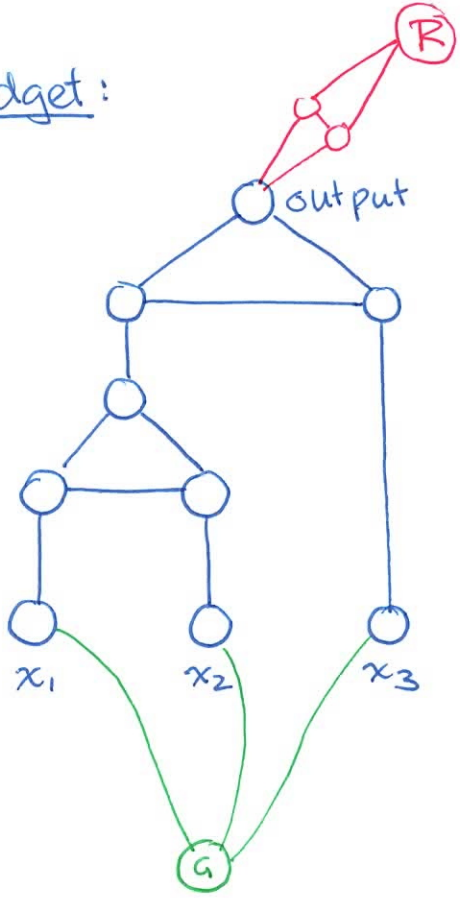


equals

negate

red = true
blue = false

Clause Gadget:



blue = false

red = true

x_1, x_2 & x_3 must be blue or red in any 3-coloring

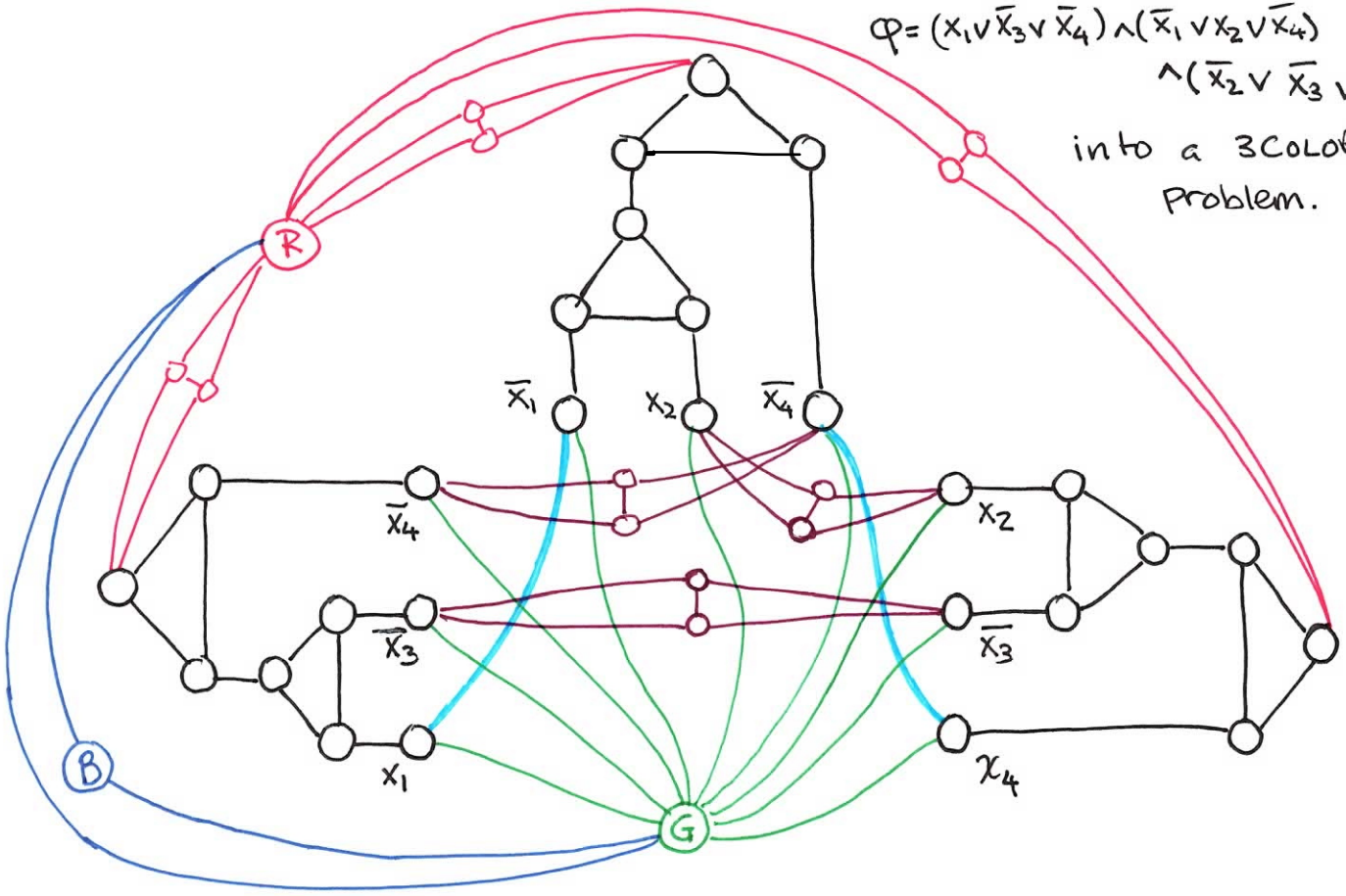
if x_1, x_2 & x_3 are all blue, output must be blue

if one of x_1, x_2 & x_3 is red, then output can be red

transform

$\varphi = (x_1 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_4)$
$\wedge (\bar{x}_2 \vee \bar{x}_3 \vee x_4)$

into a 3COLOR problem.



Recap: We can efficiently transform an instance φ of SAT into an instance G' of 3COLOR s.t.

$$\varphi \in \text{SAT} \iff G' \in \text{3COLOR}$$

I.e., we reduced SAT to 3COLOR.

Previously, we reduced 3COLOR to SAT.

$\therefore$ 3COLOR and SAT have equivalent complexity.

They are both NP-complete.

"I can't find an efficient algorithm, but neither can all these famous people."