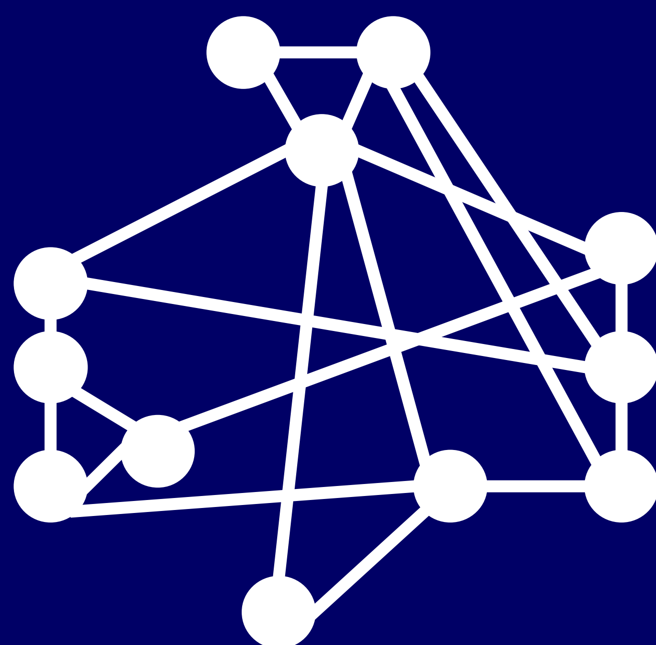
Great Theoretical Ideas In Computer Science

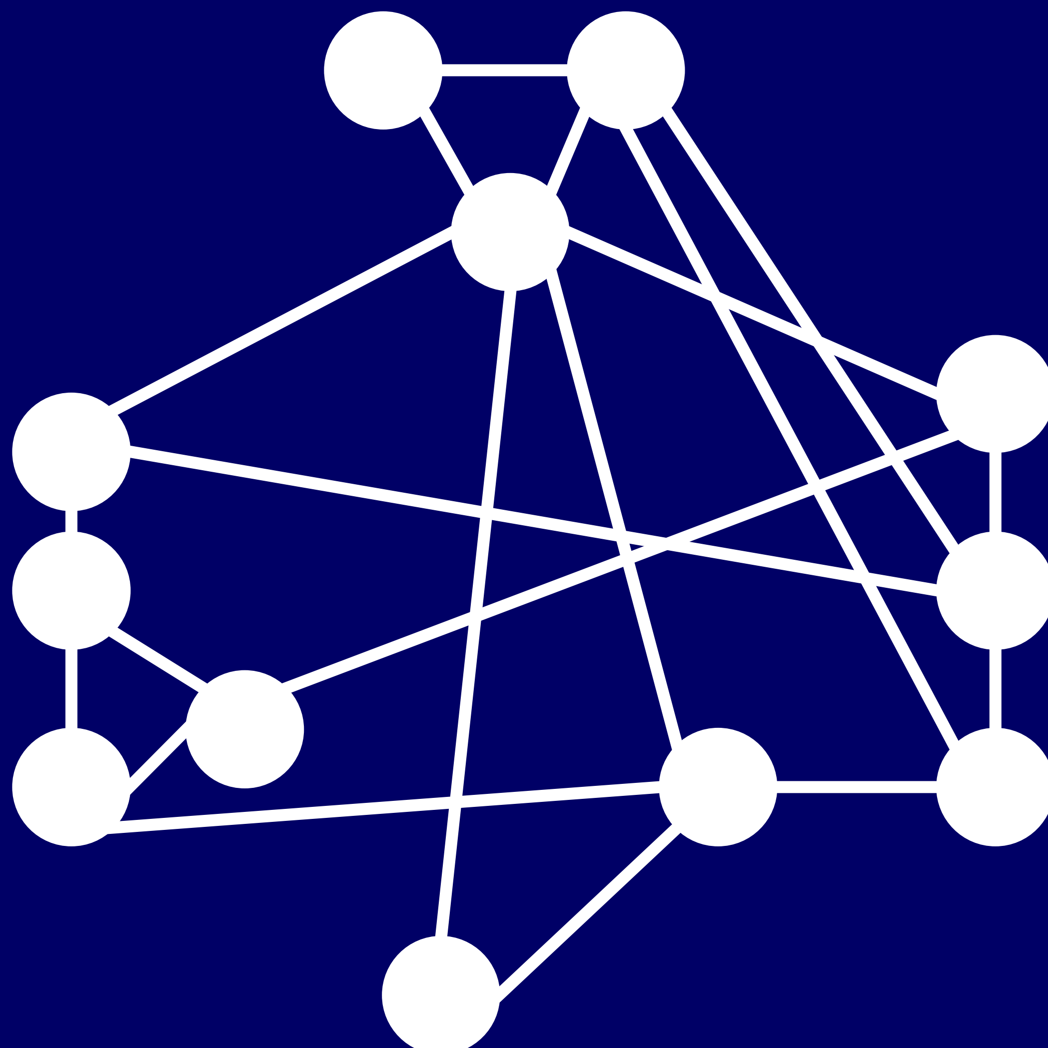
Anupam Gupta CS 15-251 Fall 2006

Lecture 28 Dec 5th, 2006 Carnegie Mellon University

Complexity Theory: A graph called "Gadget"



A Graph Named "Gadget"



K-Coloring

We define a k-coloring of a graph:

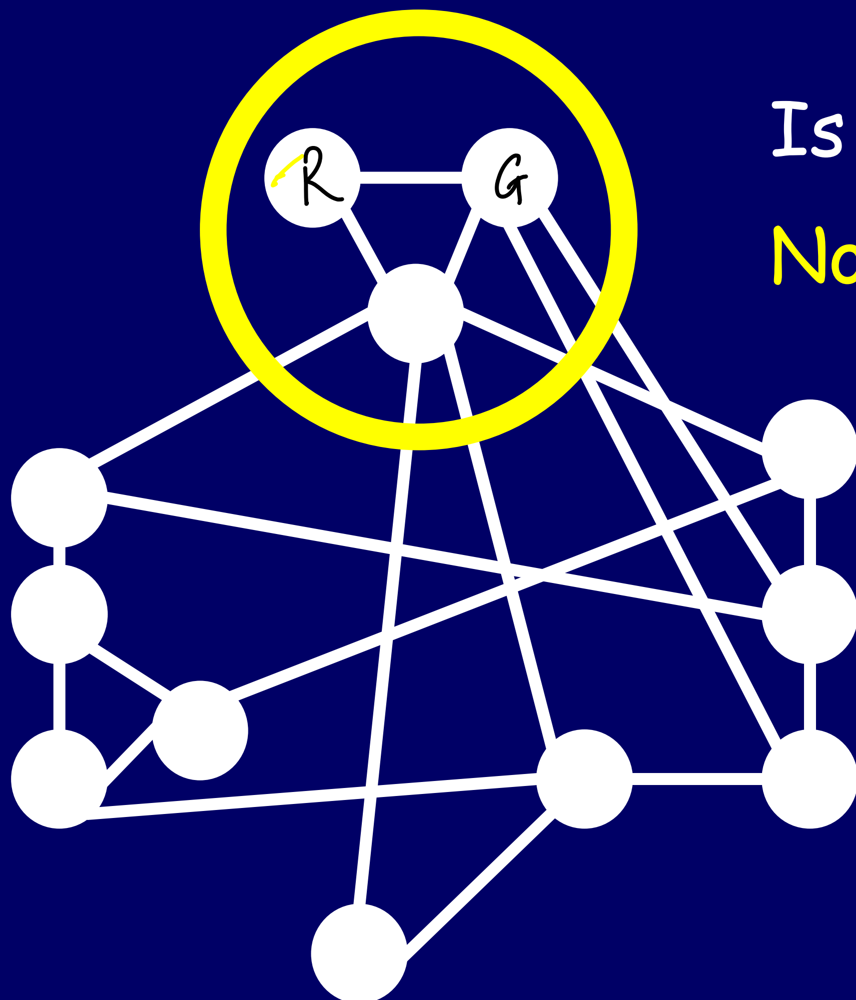
Each node gets colored with one color

At most k different colors are used

If two nodes have an edge between them they must have different colors

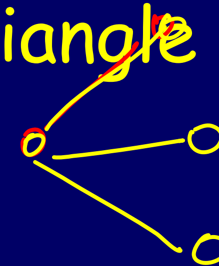
A graph is called k-colorable if and only if it has a k-coloring

A 2-CRAYOLA Question!



Is Gadget 2-colorable?

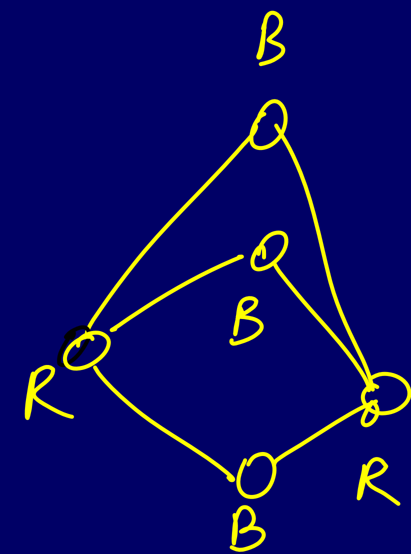
No, it contains a triangle



A 2-CRAYOLA Question!

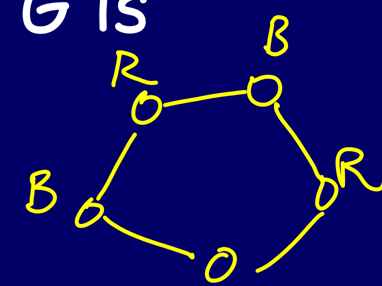
Given a graph G, how can we decide if it is 2-colorable?

Answer: Enumerate all 2^n possible colorings to look for a valid 2-color



How can we efficiently decide if G is 2-colorable?

Proposition: If G contains an odd cycle, G is not 2-colorable



Alternate coloring algorithm:

To 2-color a connected graph G, pick an arbitrary node v, and color it white

Color all v's neighbors black

Color all their uncolored neighbors white, and so on

If the algorithm terminates without a color conflict, output the 2-coloring

Else, output an odd cycle

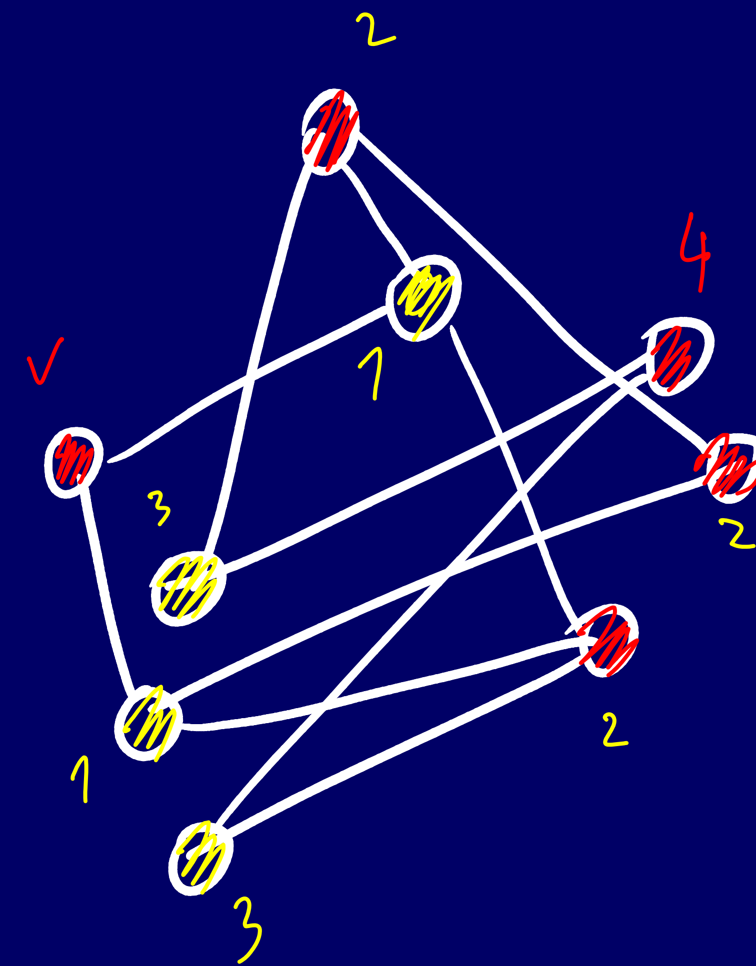
To 2-color a connected graph G, pick an arbitrary node v, and color it white

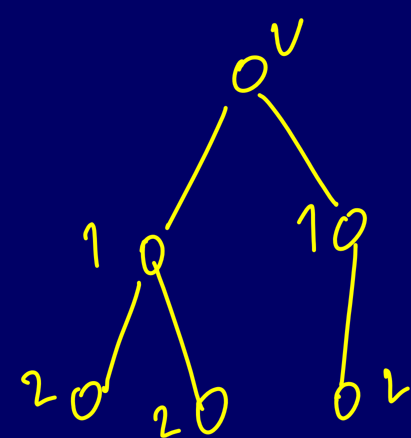
Color all v's neighbors black

Color all their uncolored neighbors white, and so on

If the algorithm terminates without a color conflict, output the 2-coloring

Else, output an odd cycle





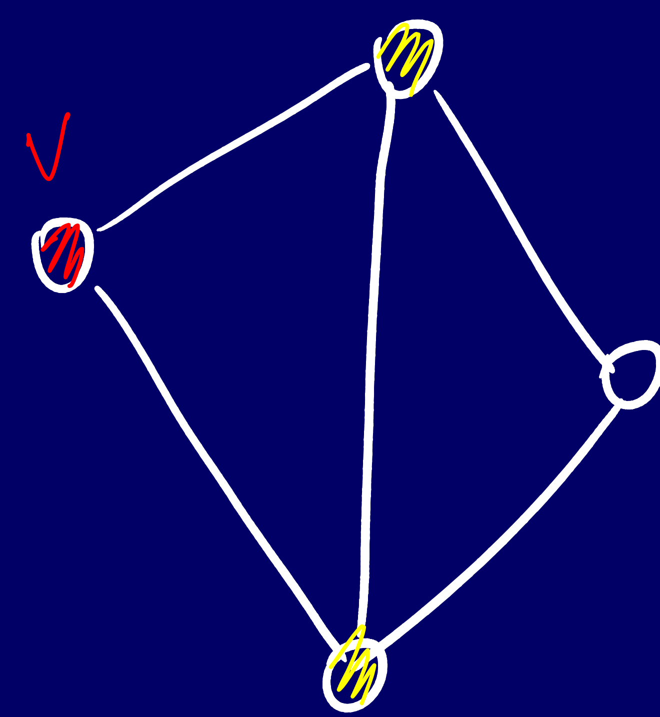
To 2-color a connected graph G, pick an arbitrary node v, and color it white

Color all v's neighbors black

Color all their uncolored neighbors white, and so on

If the algorithm terminates without a color conflict, output the 2-coloring

Else, output an odd cycle



A 2-CRAYOLA Question!

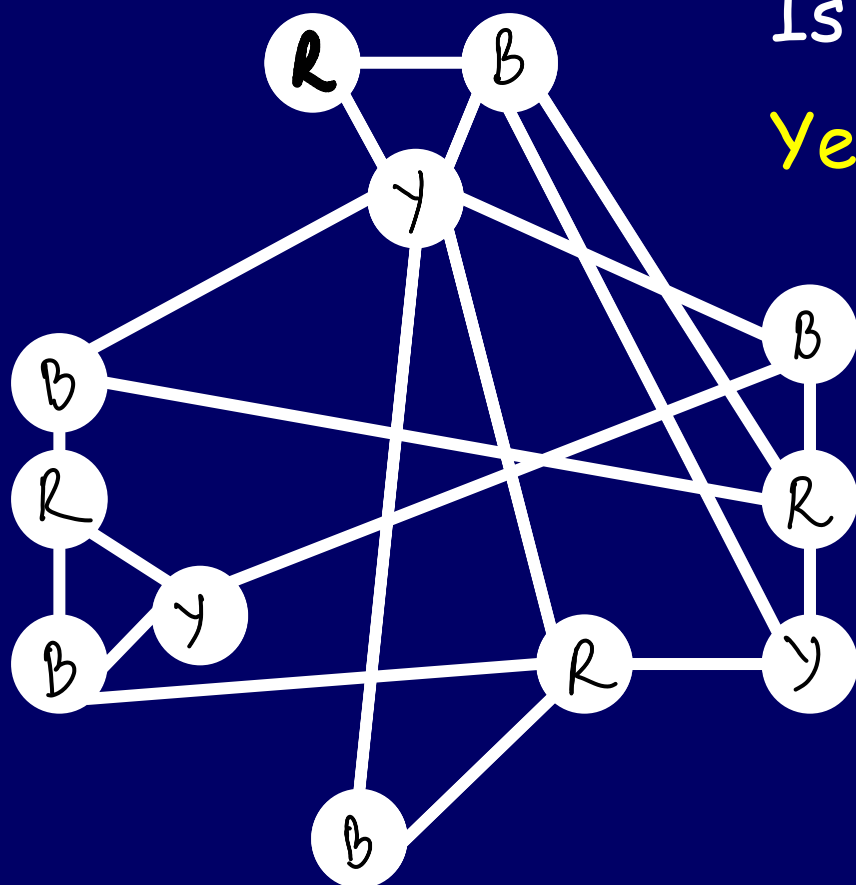
Theorem:

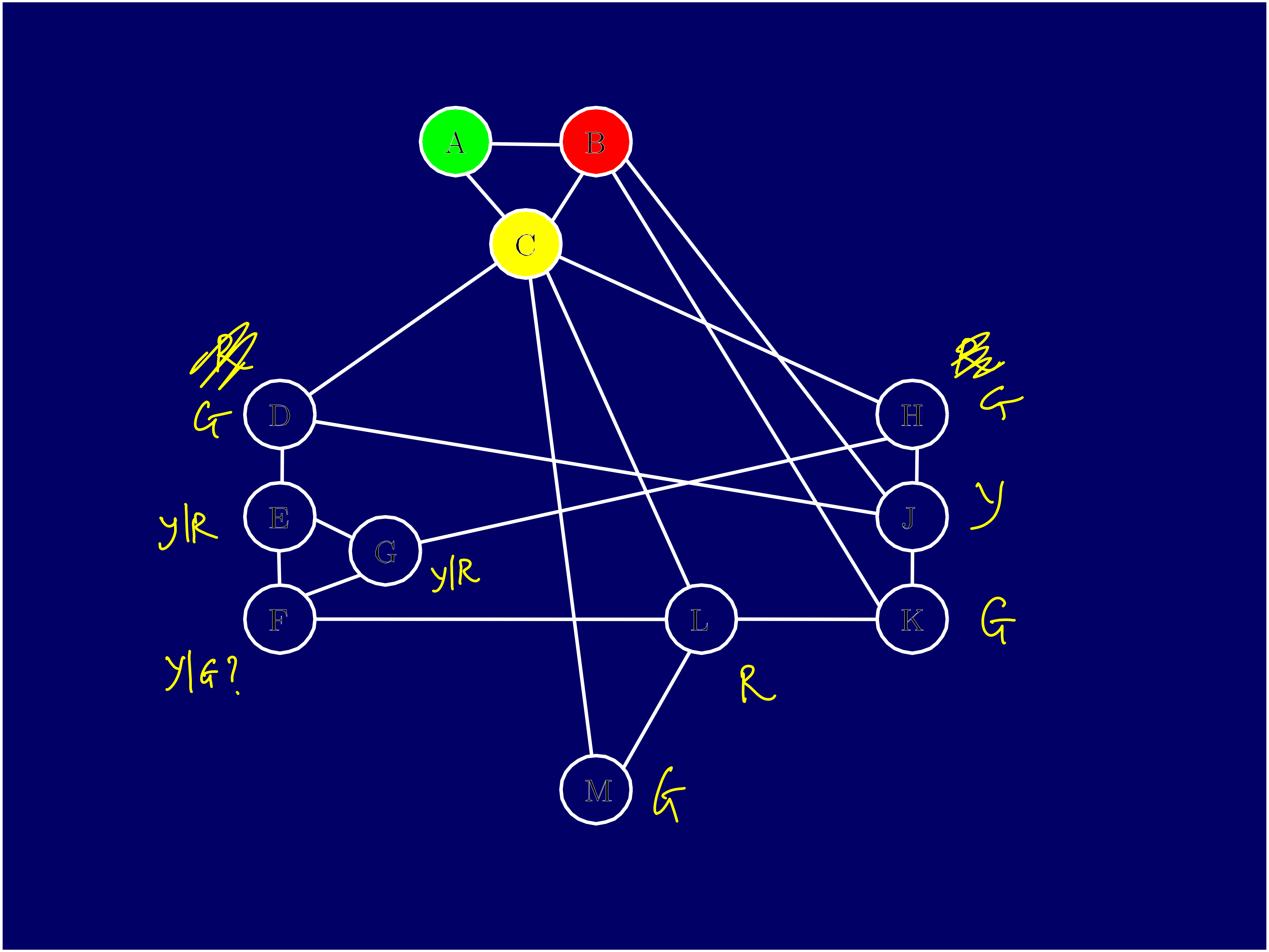
G contains an odd cycle
if and only if
G is not 2-colorable

A 3-CRAYOLA Question!

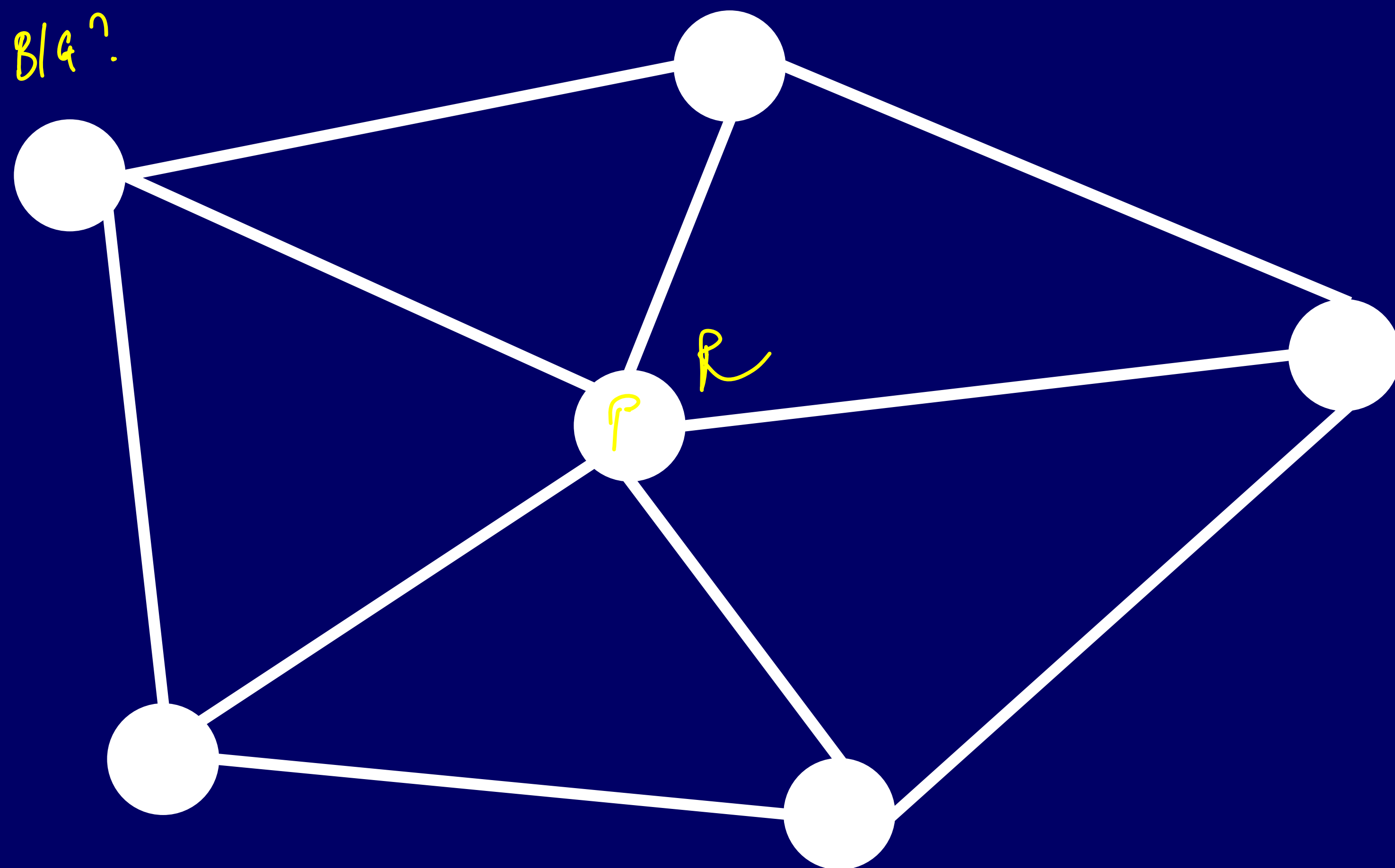
Is Gadget 3-colorable?

Yes!




A
B
C
D
E
F
G
H
J
K
L
M
G
Y|R
Y|G?
Y|R
G
Y
G
R
G

A 3-CRAYOLA Question!

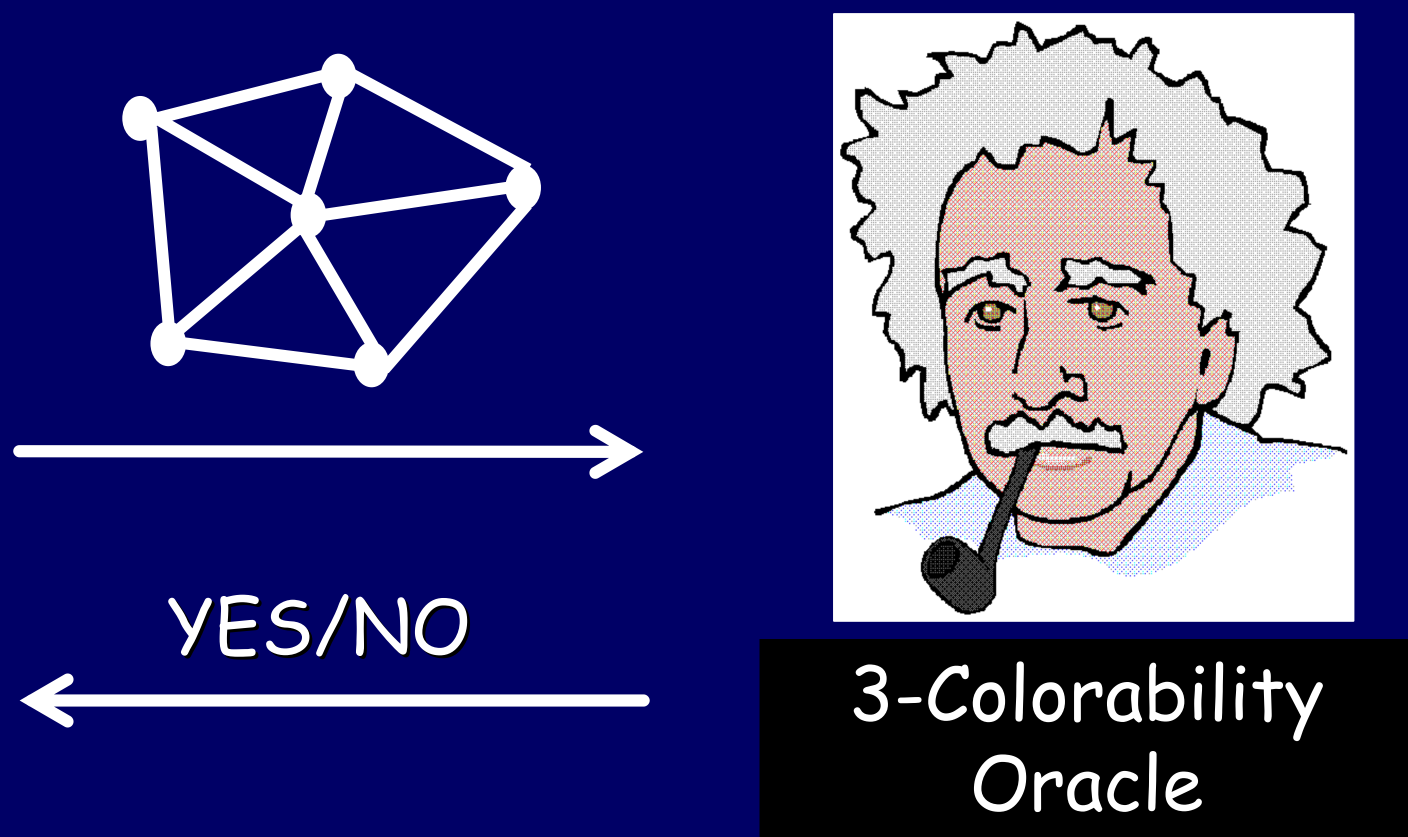


3-Coloring Is Decidable by Brute Force

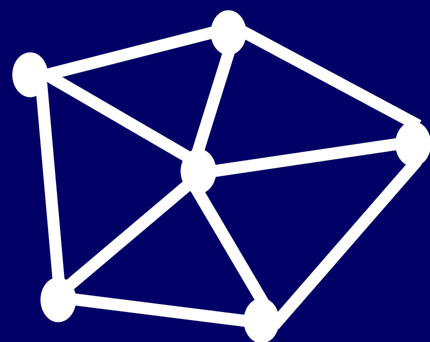
exponential(n) vs polynomial(n) n^3

Try out all 3^n colorings until you determine if G has a 3-coloring

A 3-CRAYOLA Oracle

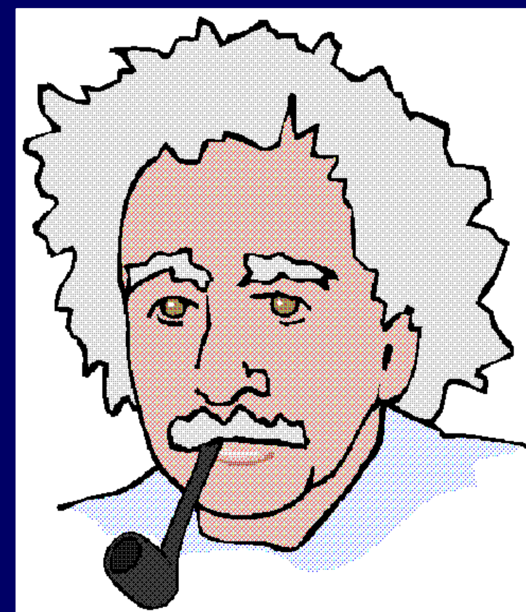


Better 3-CRAYOLA Oracle

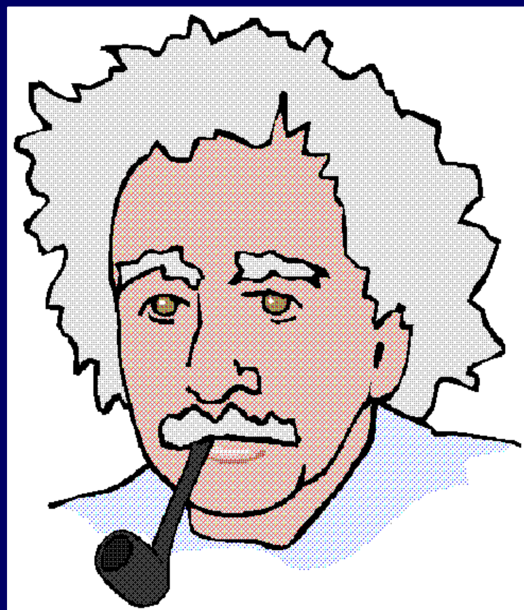


NO, or

YES, here is how: gives 3-coloring of the nodes

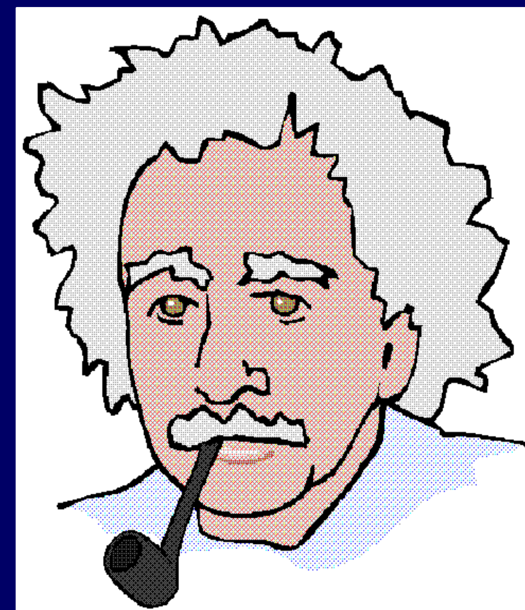


3-Colorability Search Oracle



3-Colorability Search Oracle

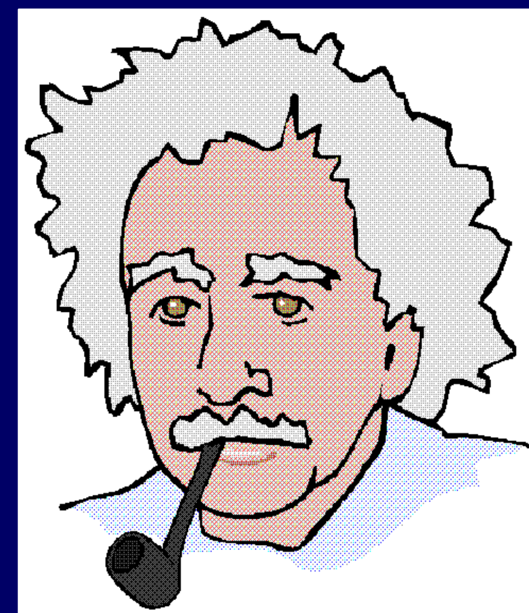
NO / Yes, here's one



3-Colorability Decision Oracle

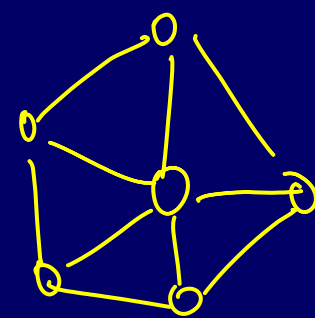
NO / YES

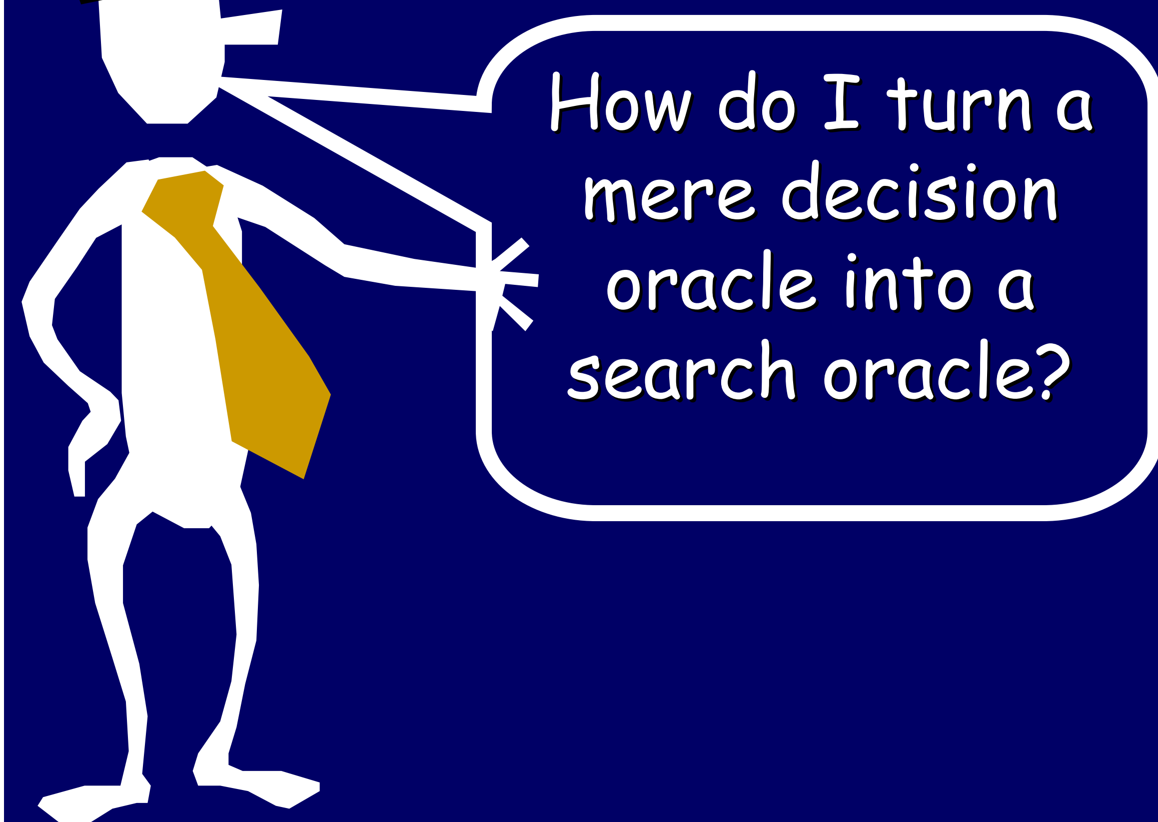
Christmas Present

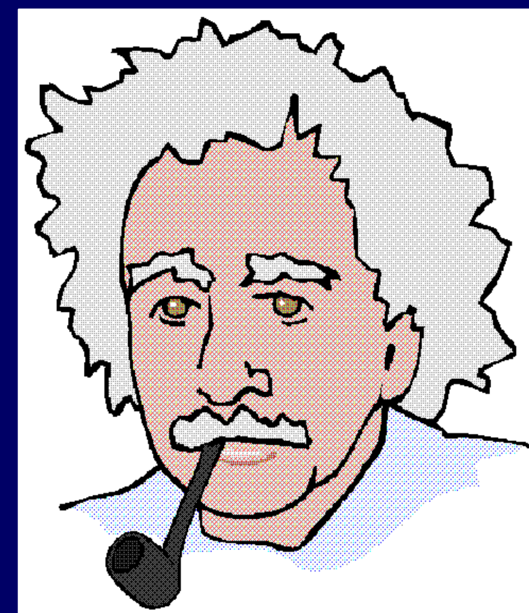


GIVEN:
3-Colorability
Decision Oracle

Christmas Present

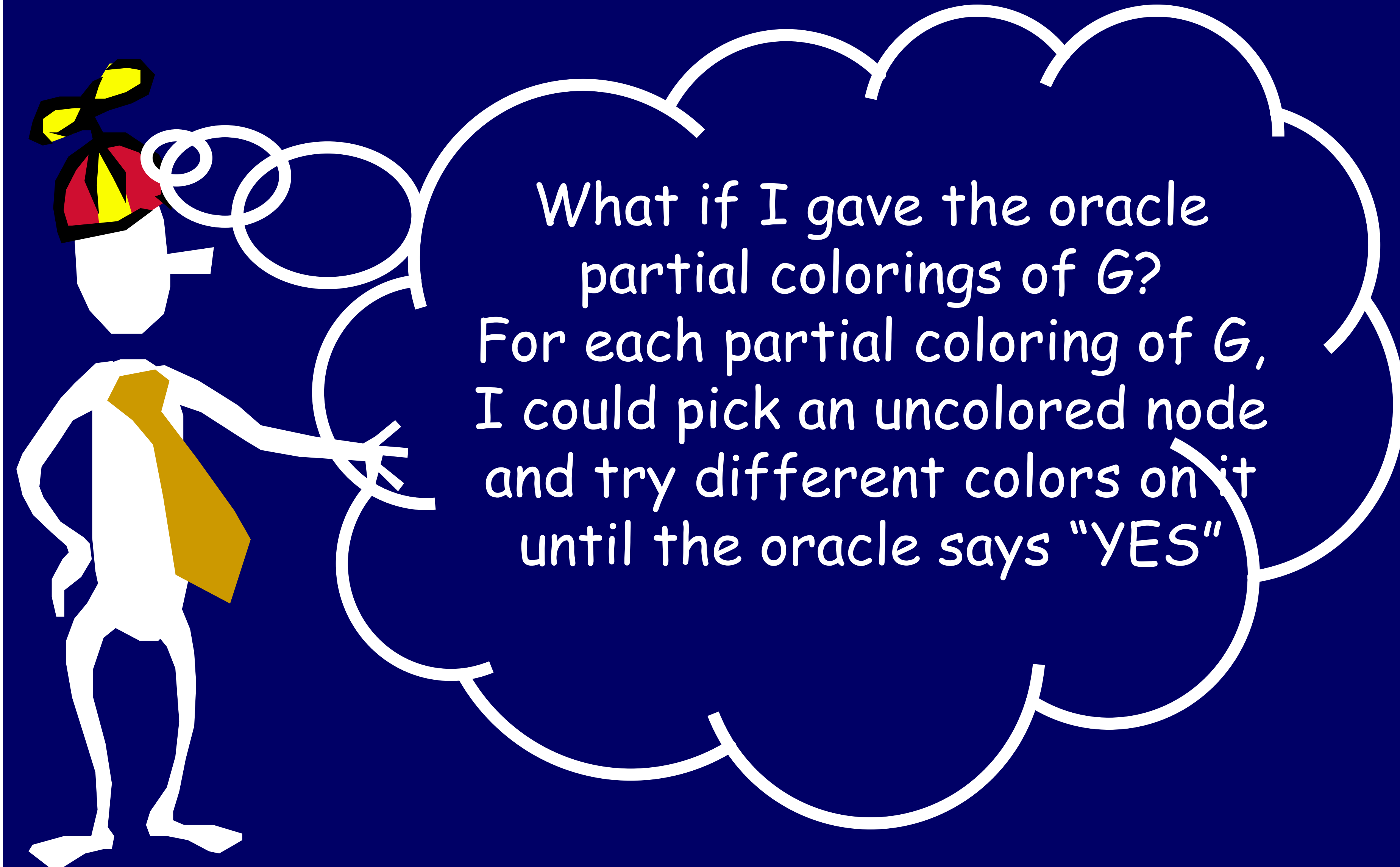




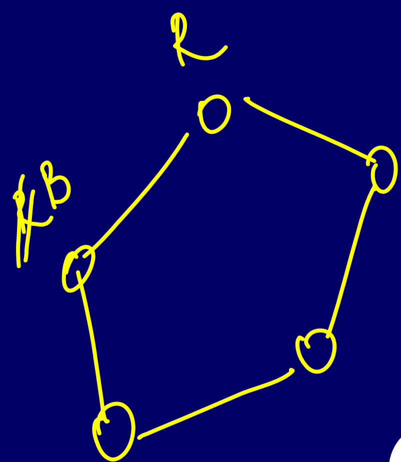


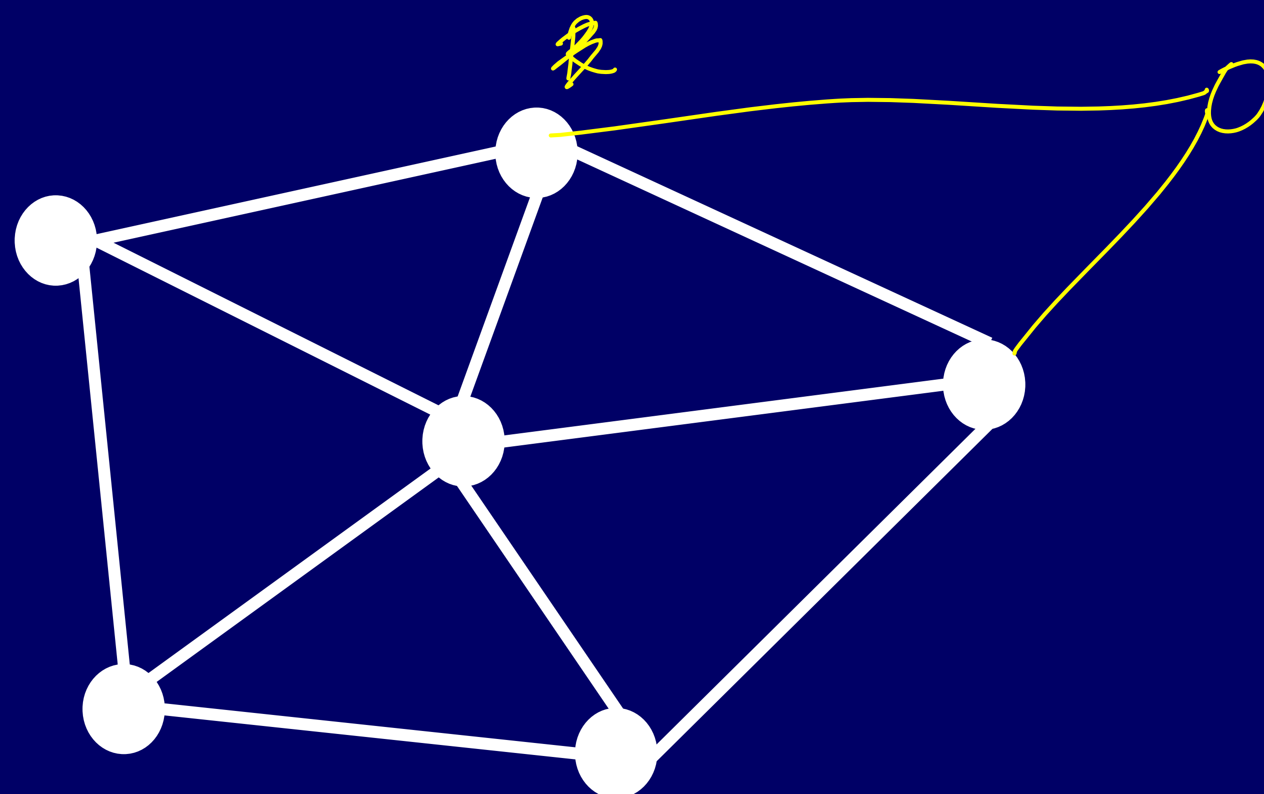
GIVEN:
3-Colorability
Decision Oracle

Beanie's Flawed Idea

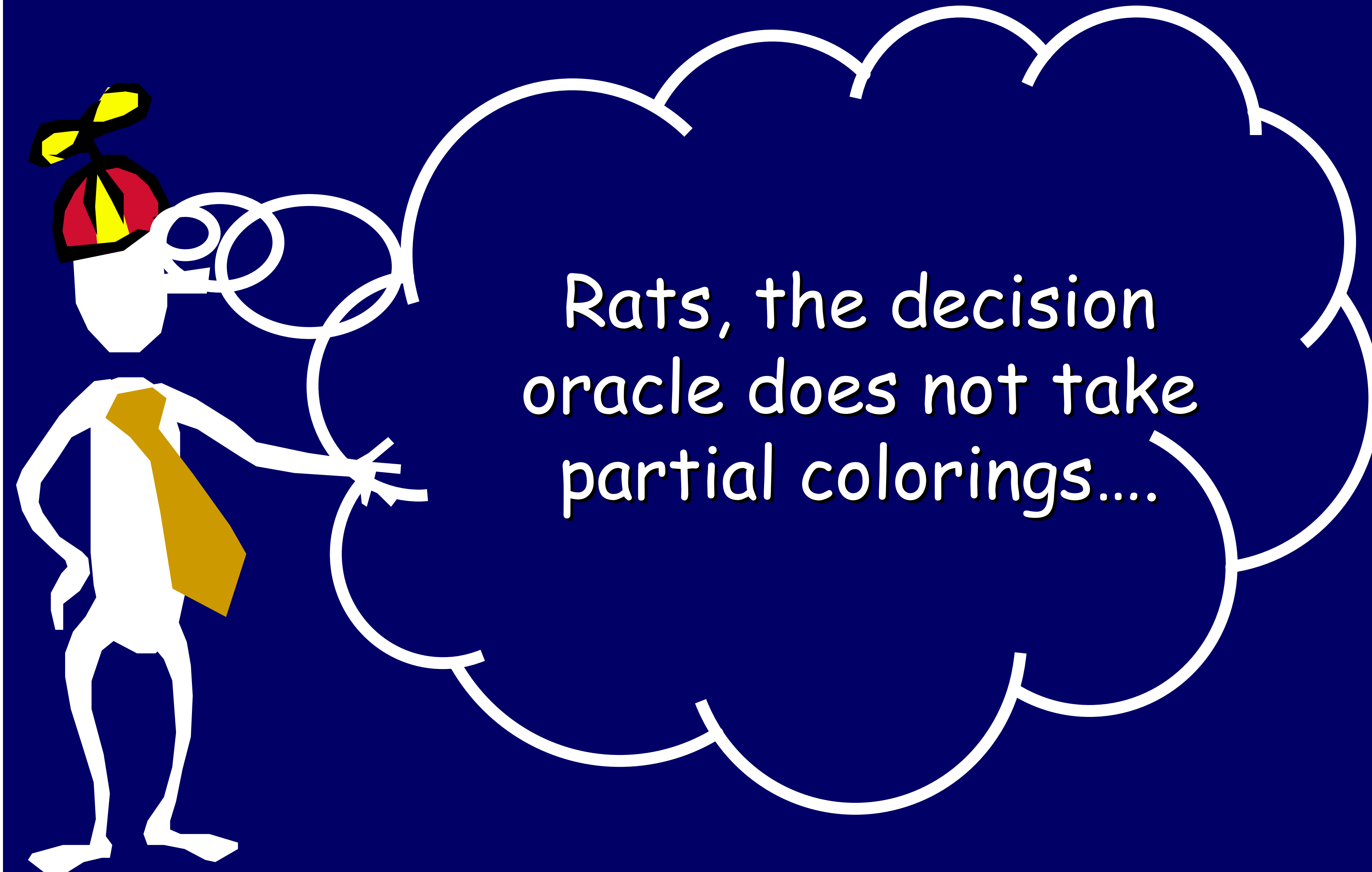


Turning a decision oracle into a search oracle

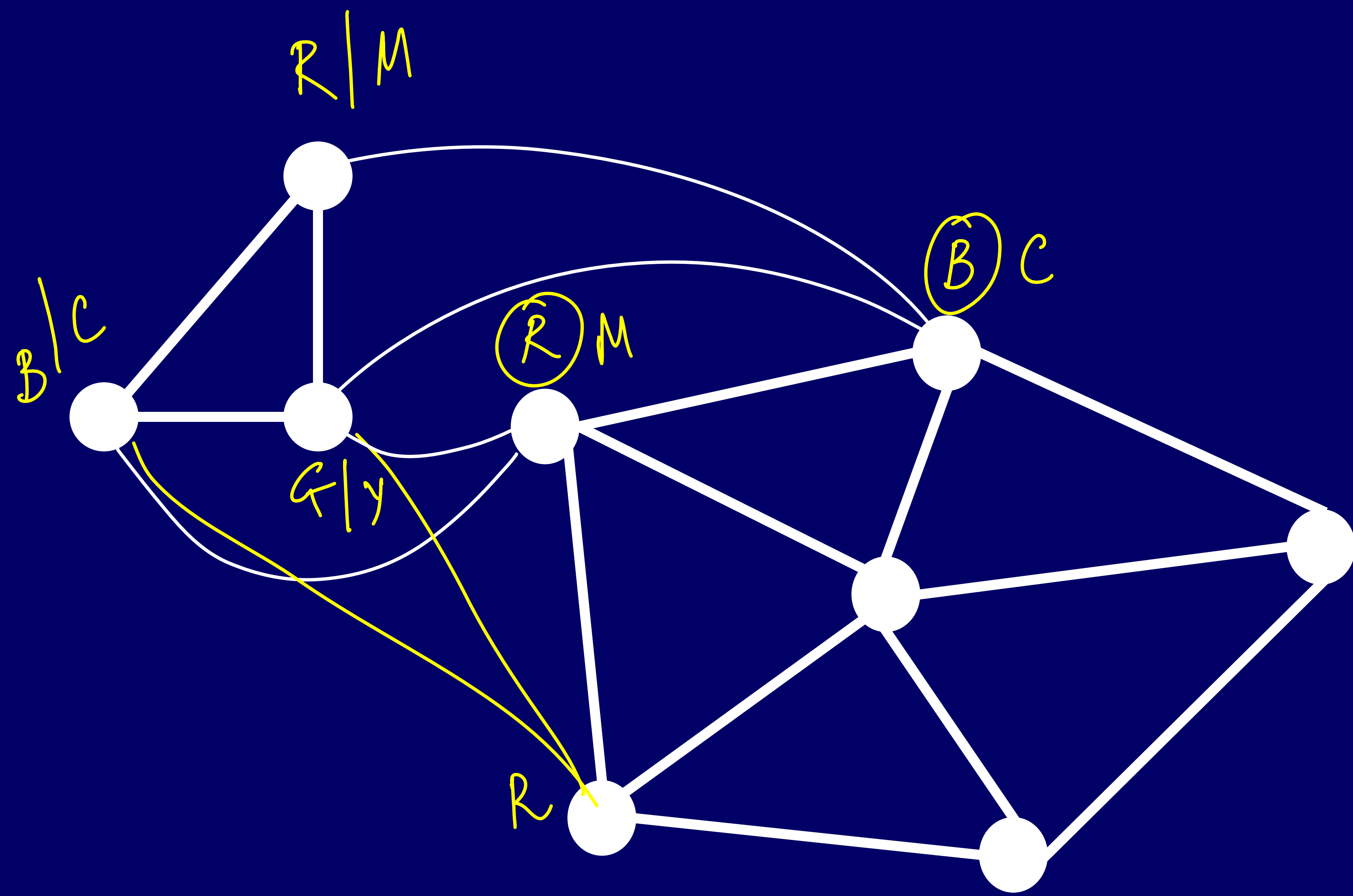




Beanie's Flawed Idea

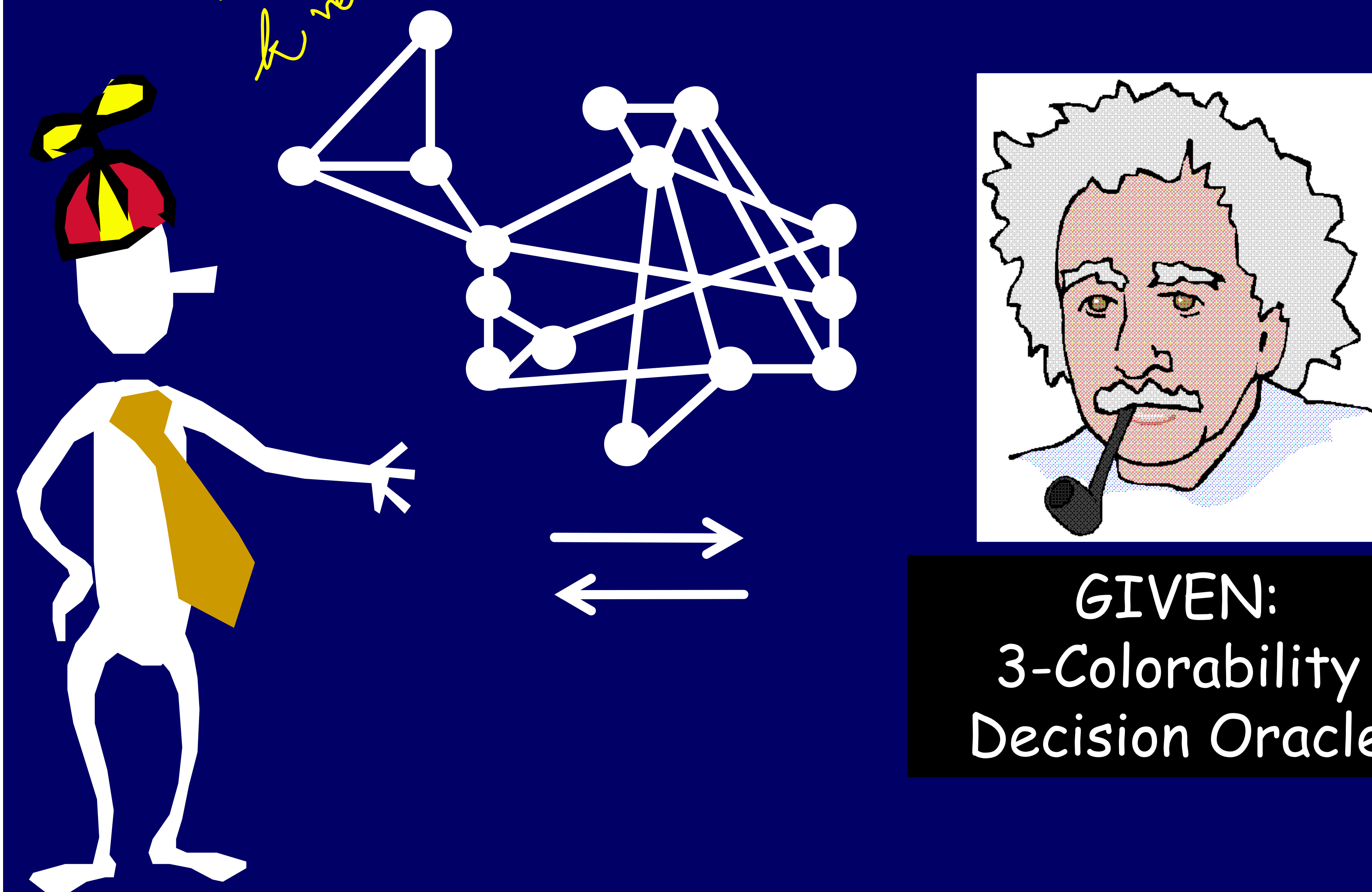


The Fix



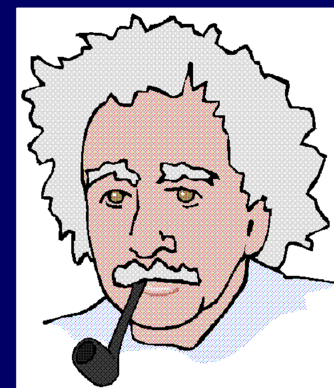
Beanie's Fix

$k=3$ vertices



GIVEN:
3-Colorability
Decision Oracle

3-Colorability
Search Oracle



3-Colorability
Decision Oracle

3^x.

$3 \cdot n$
$k \cdot n$ calls.

1 call

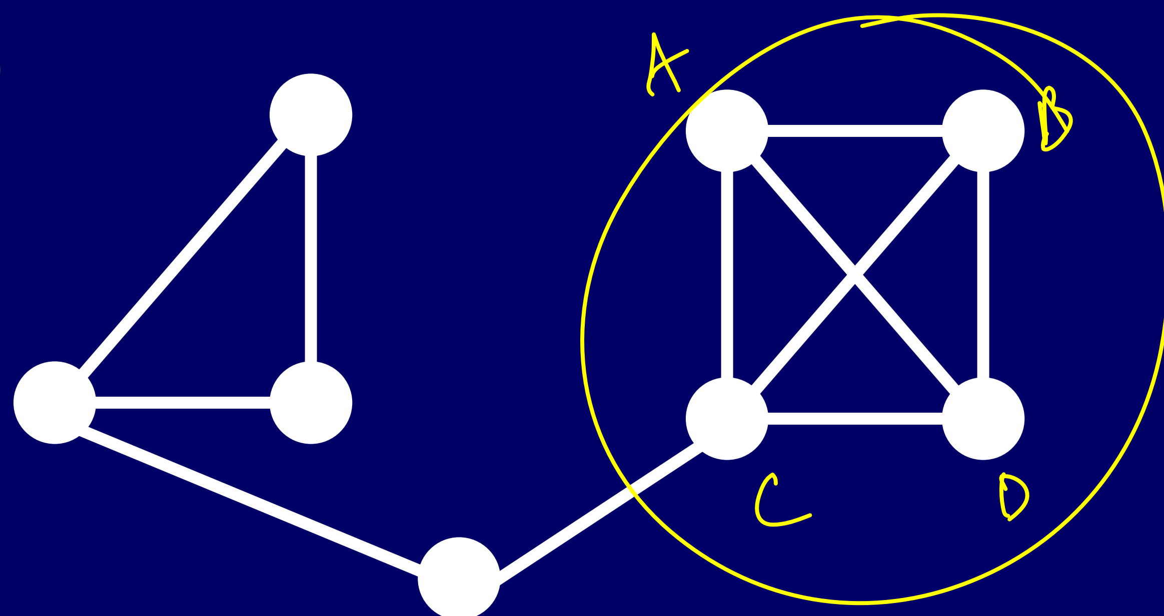
Let's now look at two other problems:

1. K-Clique
2. K-Independent Set

K-Cliques

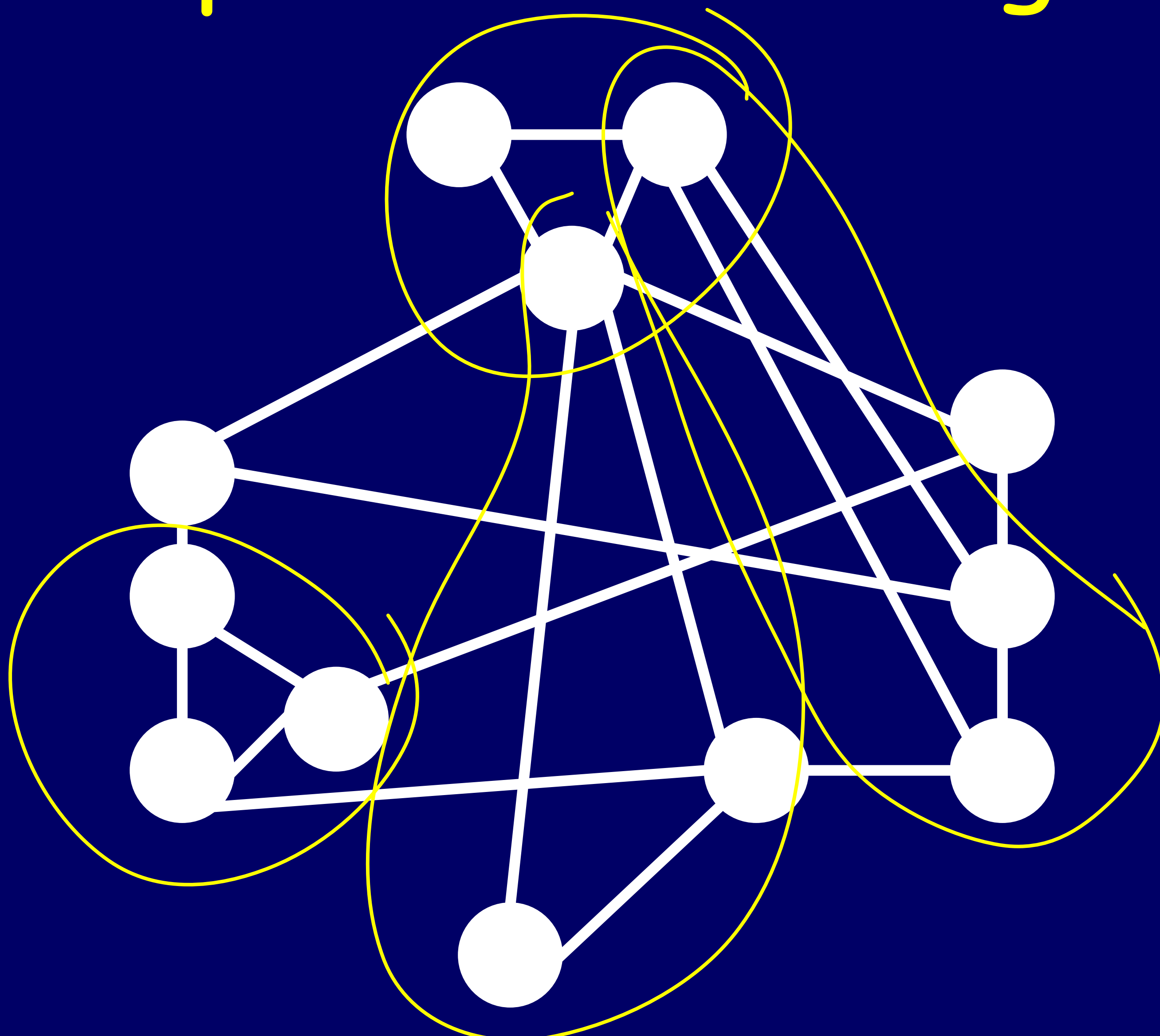
A K-clique is a set of K nodes with all K(K-1)/2 possible edges between them

$\binom{K}{2}$



This graph contains a 4-clique

A Graph Named "Gadget"

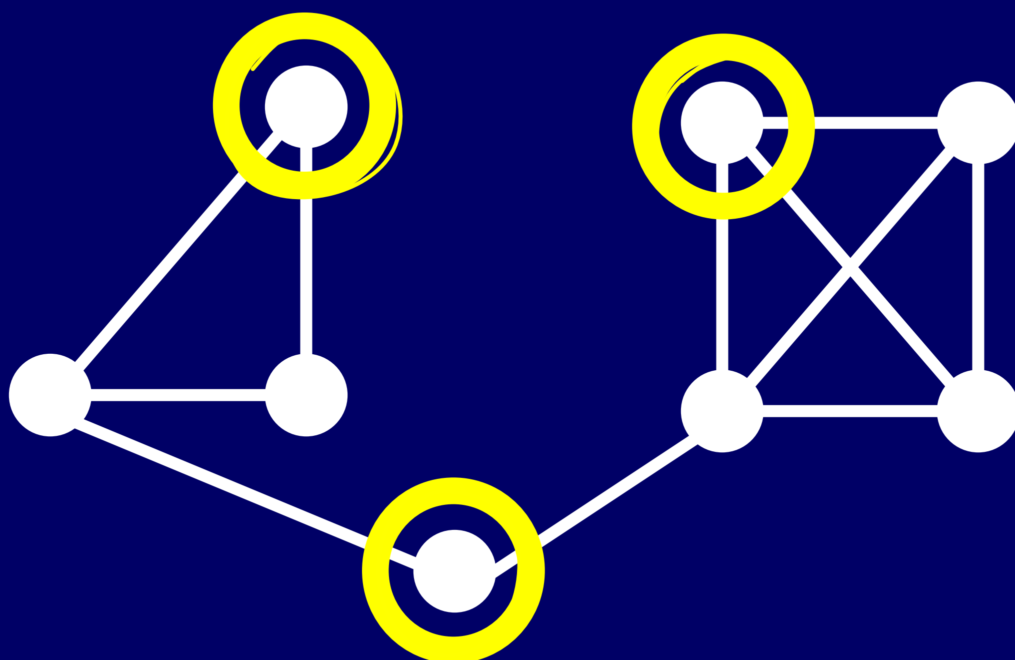


Given: (G, k)
Question: Does G contain a k-clique?

BRUTE FORCE: Try out all {n choose k} possible locations for the k clique

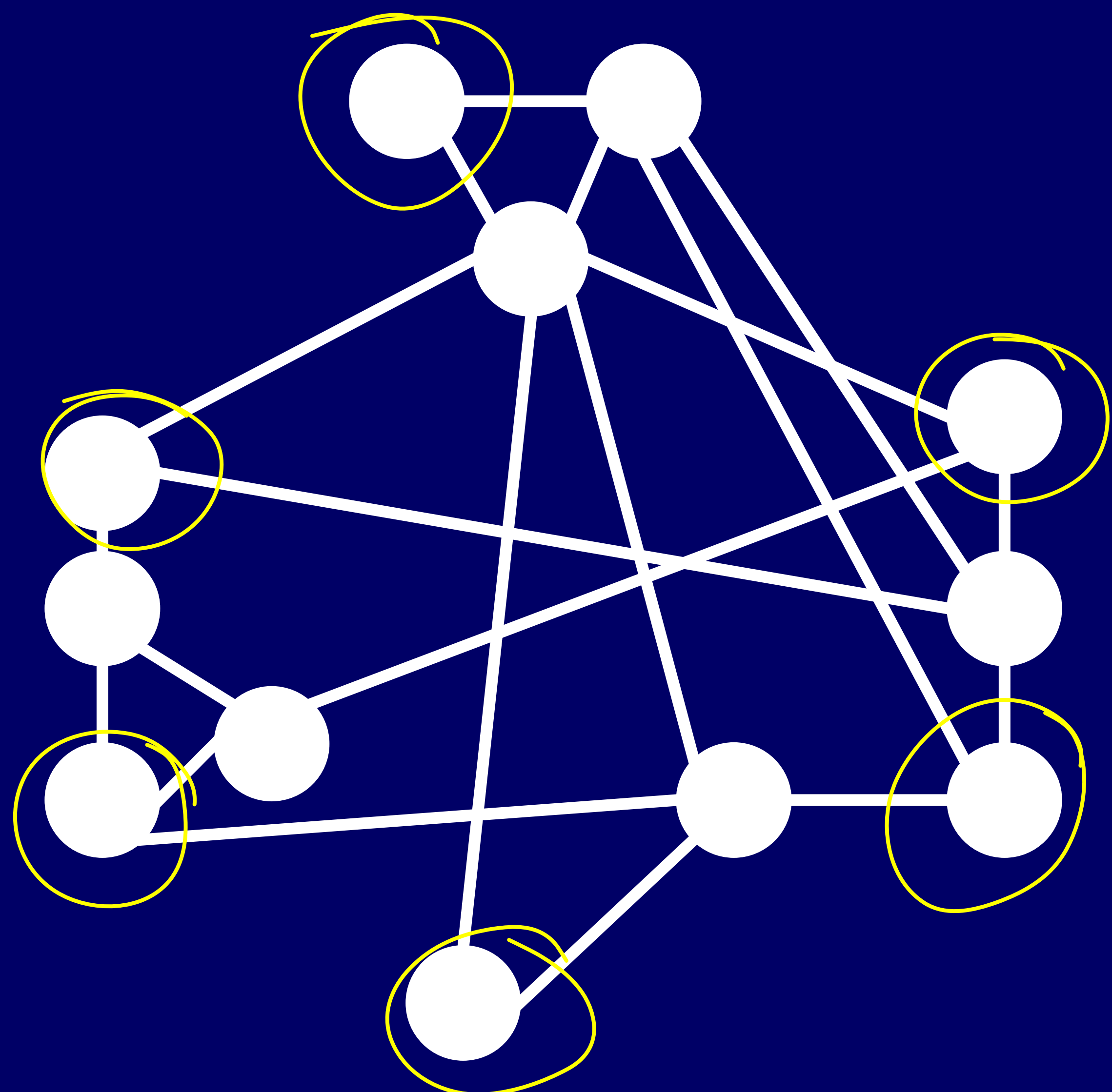
Independent Set

An independent set is a set of nodes with no edges between them



This graph contains an independent set of size 3

A Graph Named "Gadget"



Given: (G, k)
Question: Does G contain an independent set of size k?

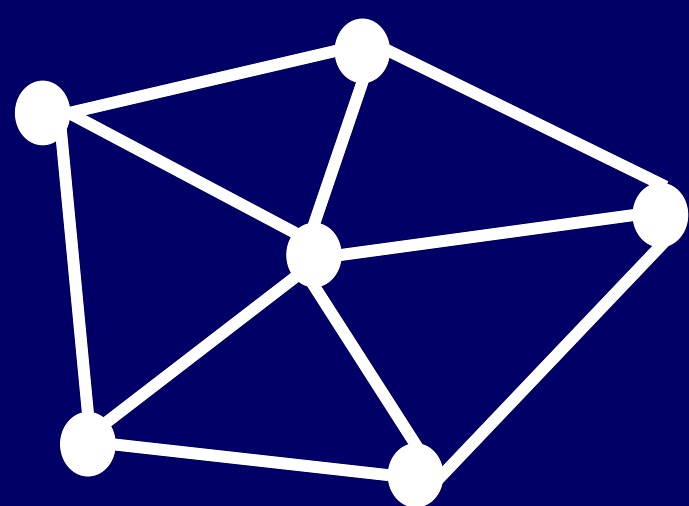
BRUTE FORCE: Try out all n choose k possible locations for the k independent set

Clique / Independent Set

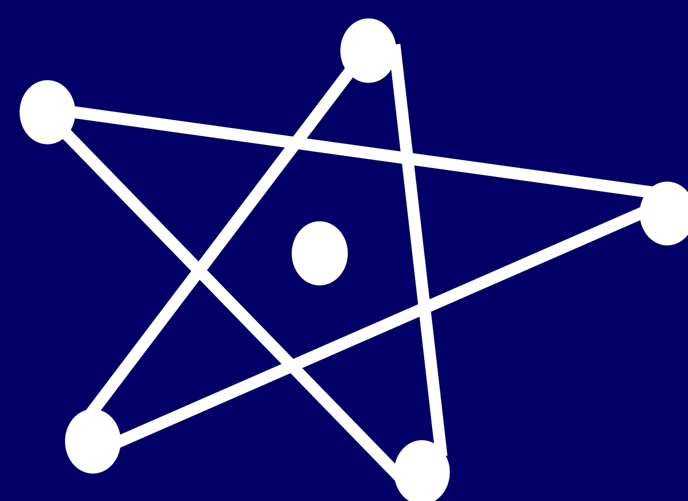
Two problems that are cosmetically different, but substantially the same

Complement of G

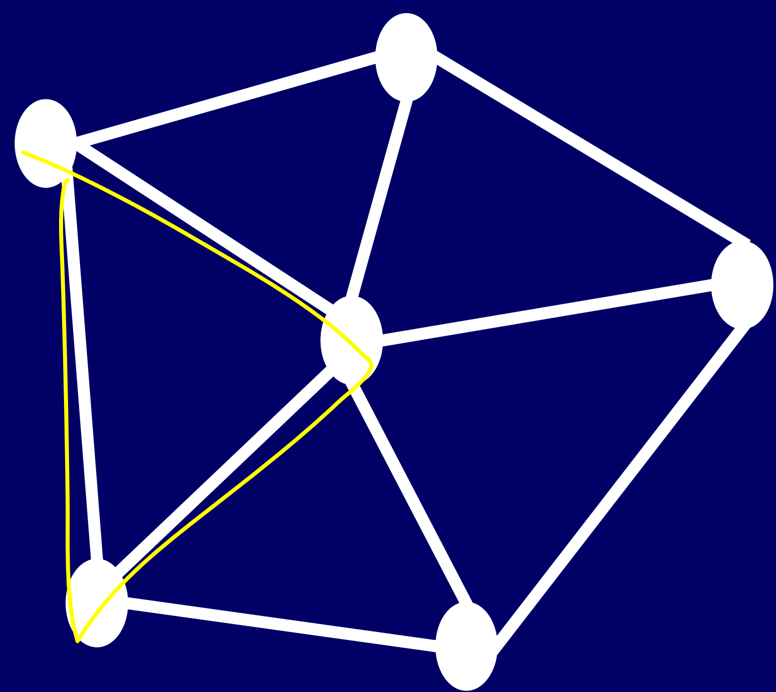
Given a graph G, let G*, the complement of G, be the graph such that
two nodes are connected in G* if and only if the corresponding nodes are not connected in G

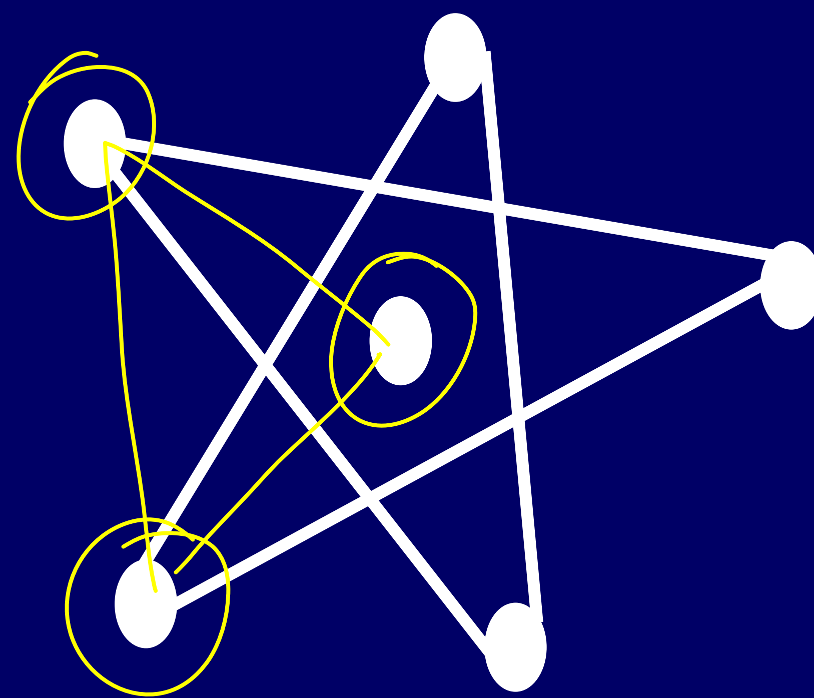


G



G*



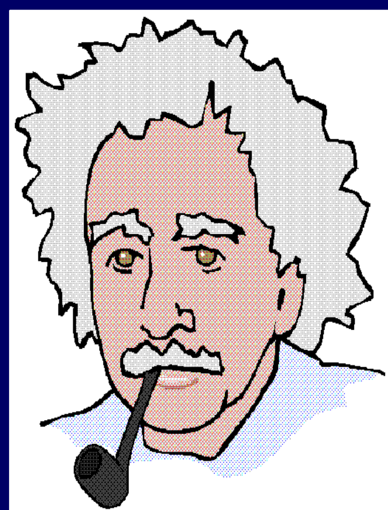


G has a k-clique $\Leftrightarrow$ G^* has an independent set of size k

Let G be an n-node graph

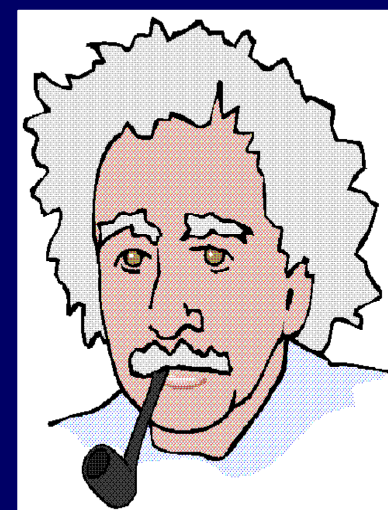
(G,k)

YES/NO



(G^*, k)

YES/NO

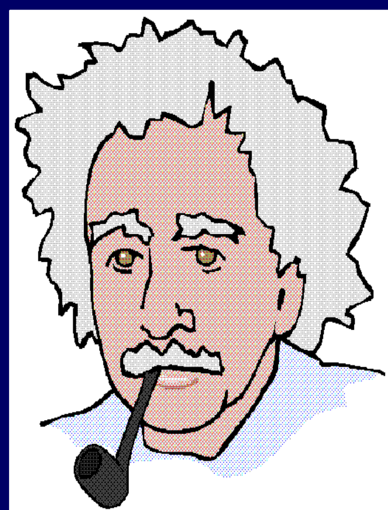


BUILD: Independent Set Oracle

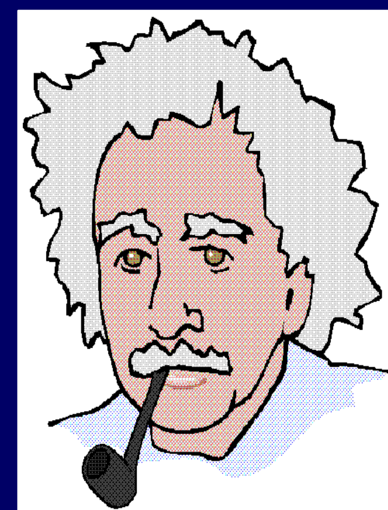
GIVEN: Clique Oracle

Let G be an n-node graph

(G,k)



(G^*, k)



BUILD:
Clique
Oracle

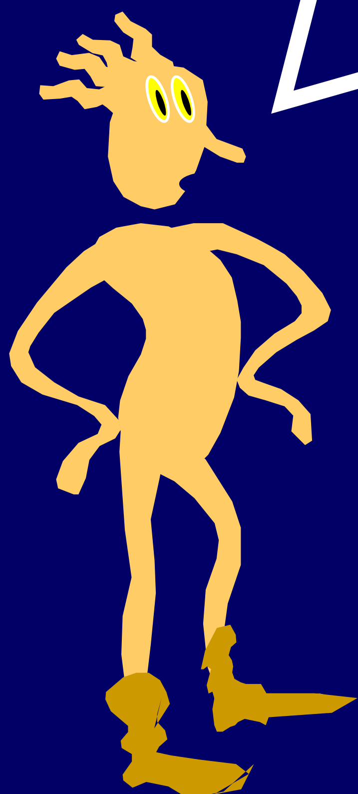
GIVEN:
Independent
Set Oracle

Clique / Independent Set

Two problems that are cosmetically different, but substantially the same

Thus, we can quickly
reduce a clique problem
to an independent set
problem and vice versa

There is a fast
algorithm to solve one
if and only if
there is a fast
algorithm for the other

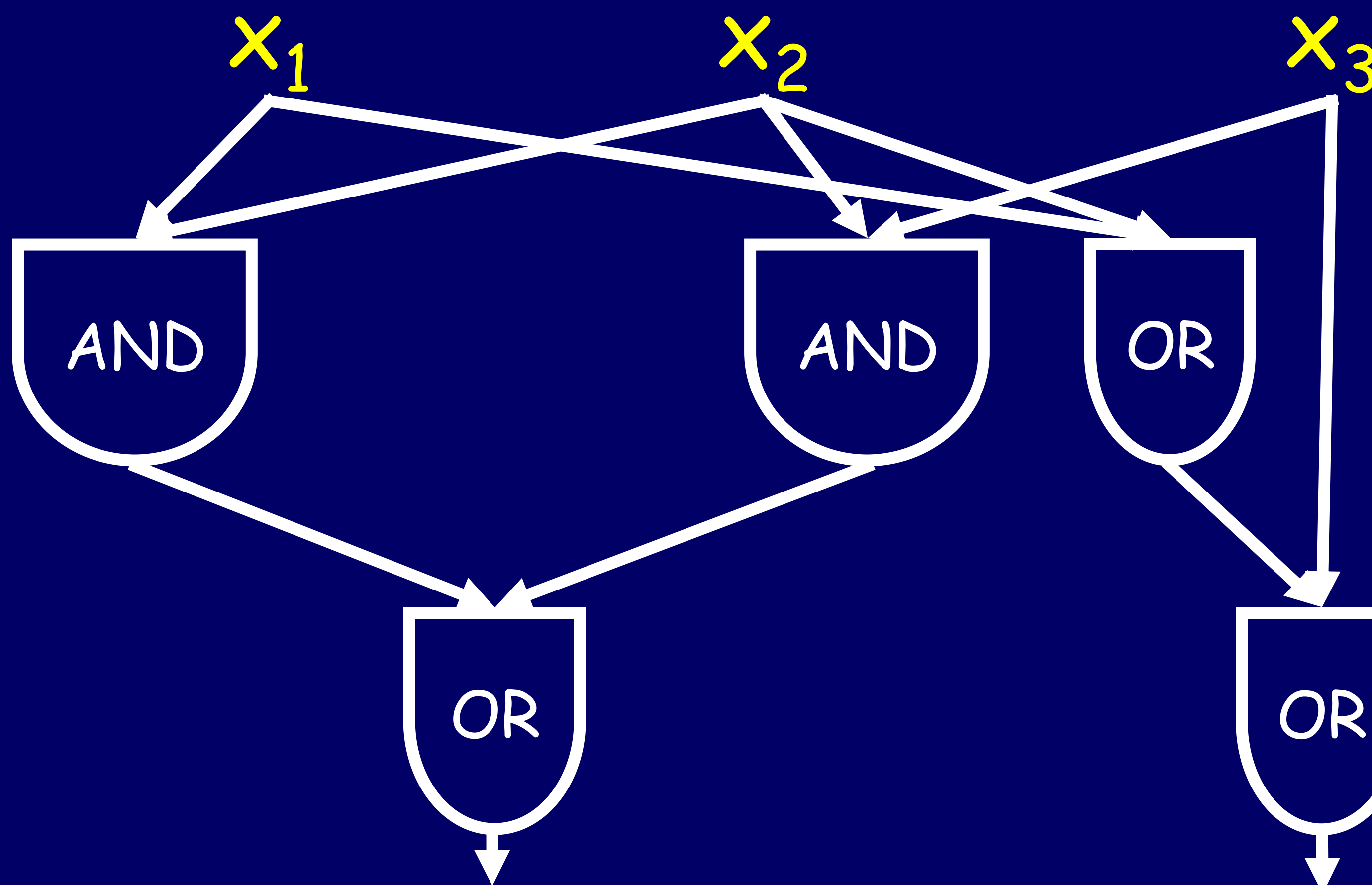


Let's now look at two other problems:

1. Circuit Satisfiability
2. Graph 3-Colorability

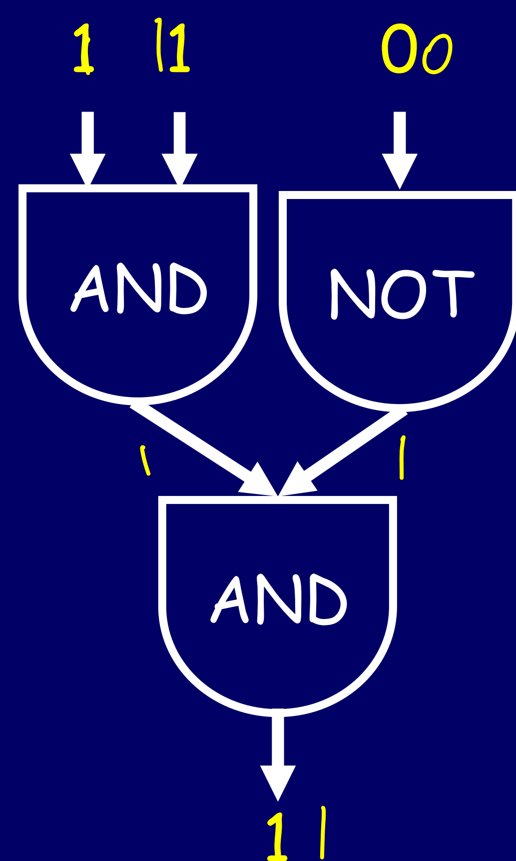
Combinatorial Circuits

AND, OR, NOT, 0, 1 gates wired together with no feedback allowed



Circuit-Satisfiability

Given a circuit with n-inputs and one output, is there a way to assign 0-1 values to the input wires so that the output value is 1 (true)?



Yes, this circuit is satisfiable: 110

Circuit-Satisfiability

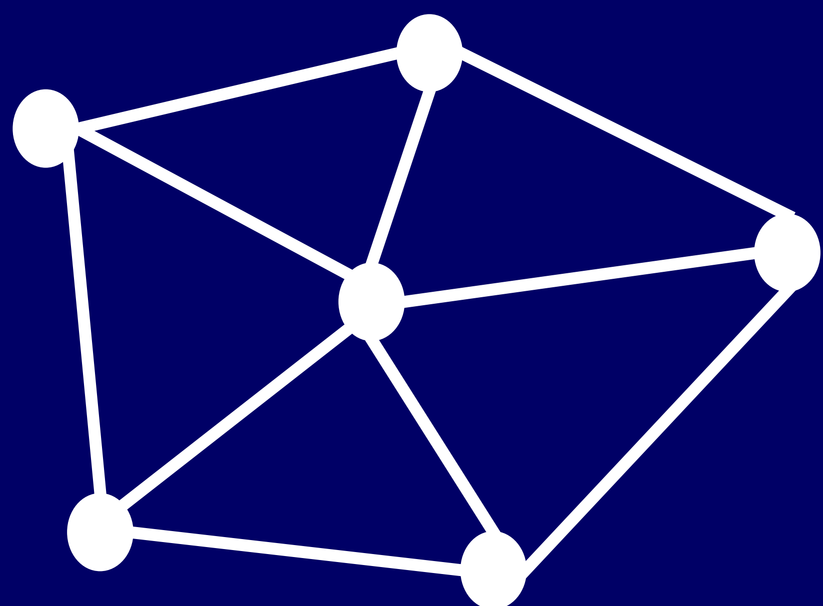
Given: A circuit with n-inputs and one output, is there a way to assign 0-1 values to the input wires so that the output value is 1 (true)?

BRUTE FORCE: Try out all 2^n assignments

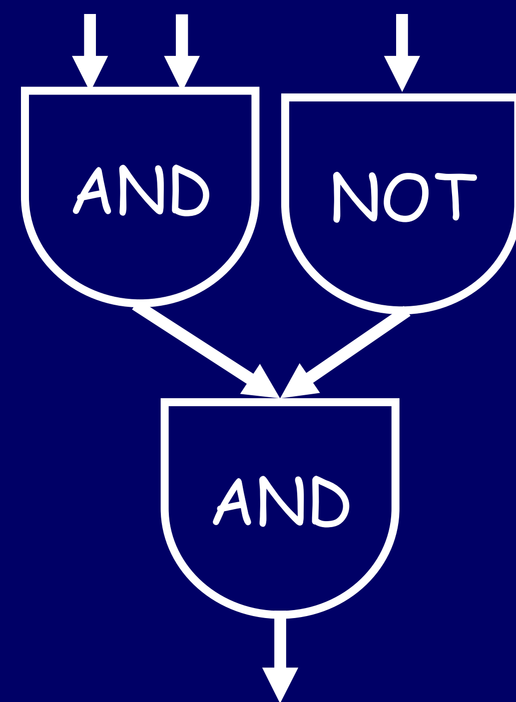
polynomial(n)

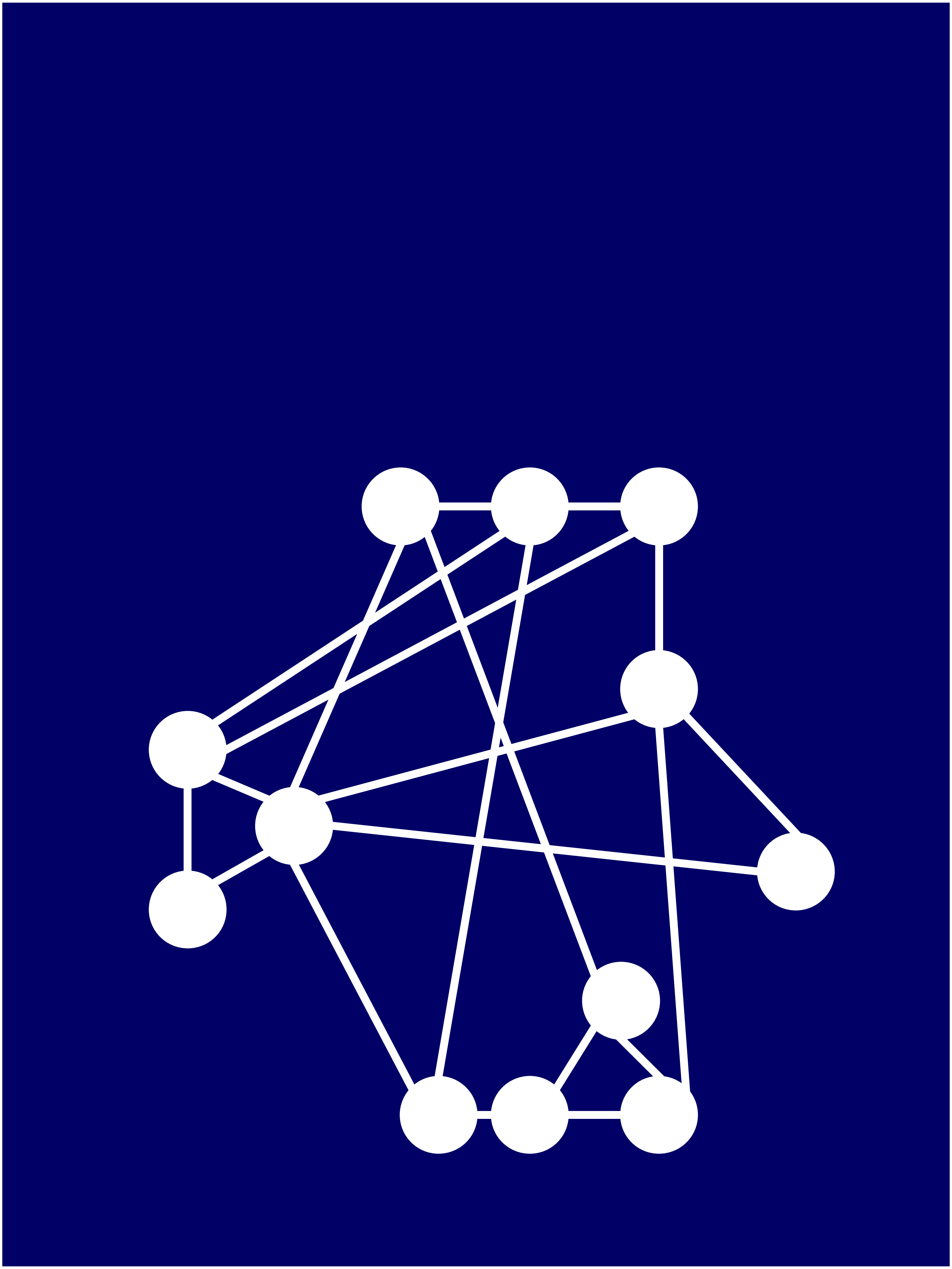
$1 Million Questions

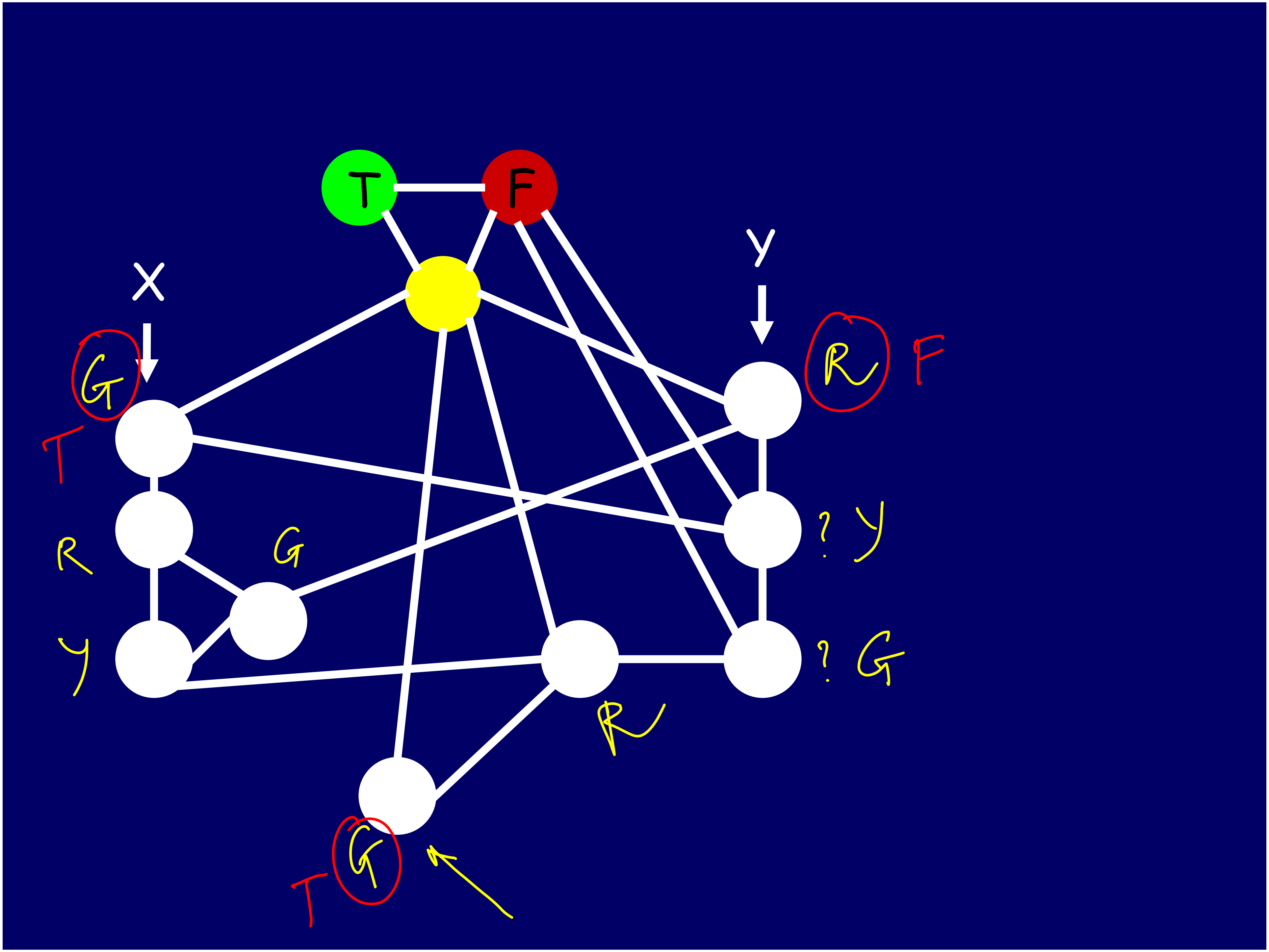
3-Colorability



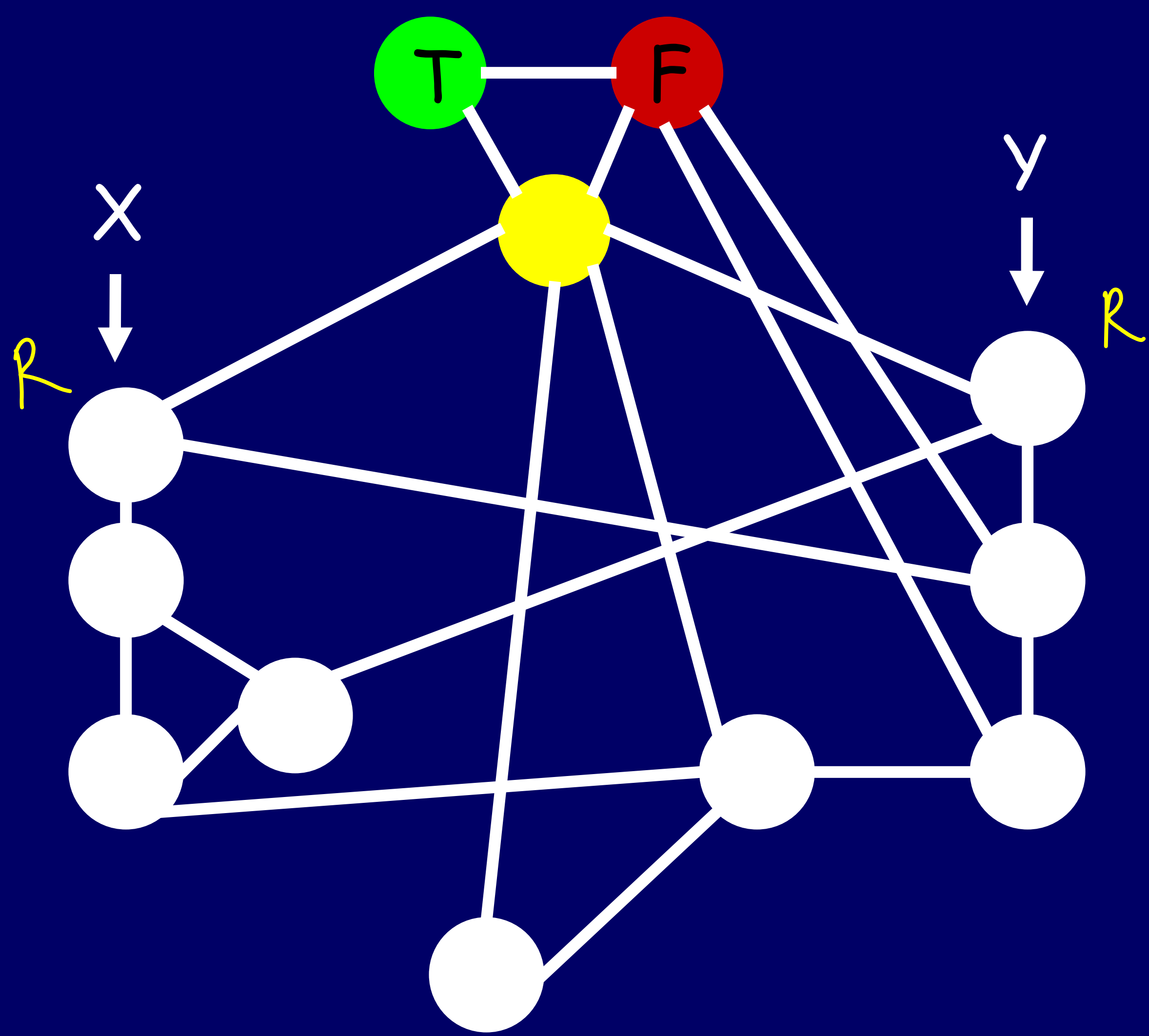
Circuit Satisfiability



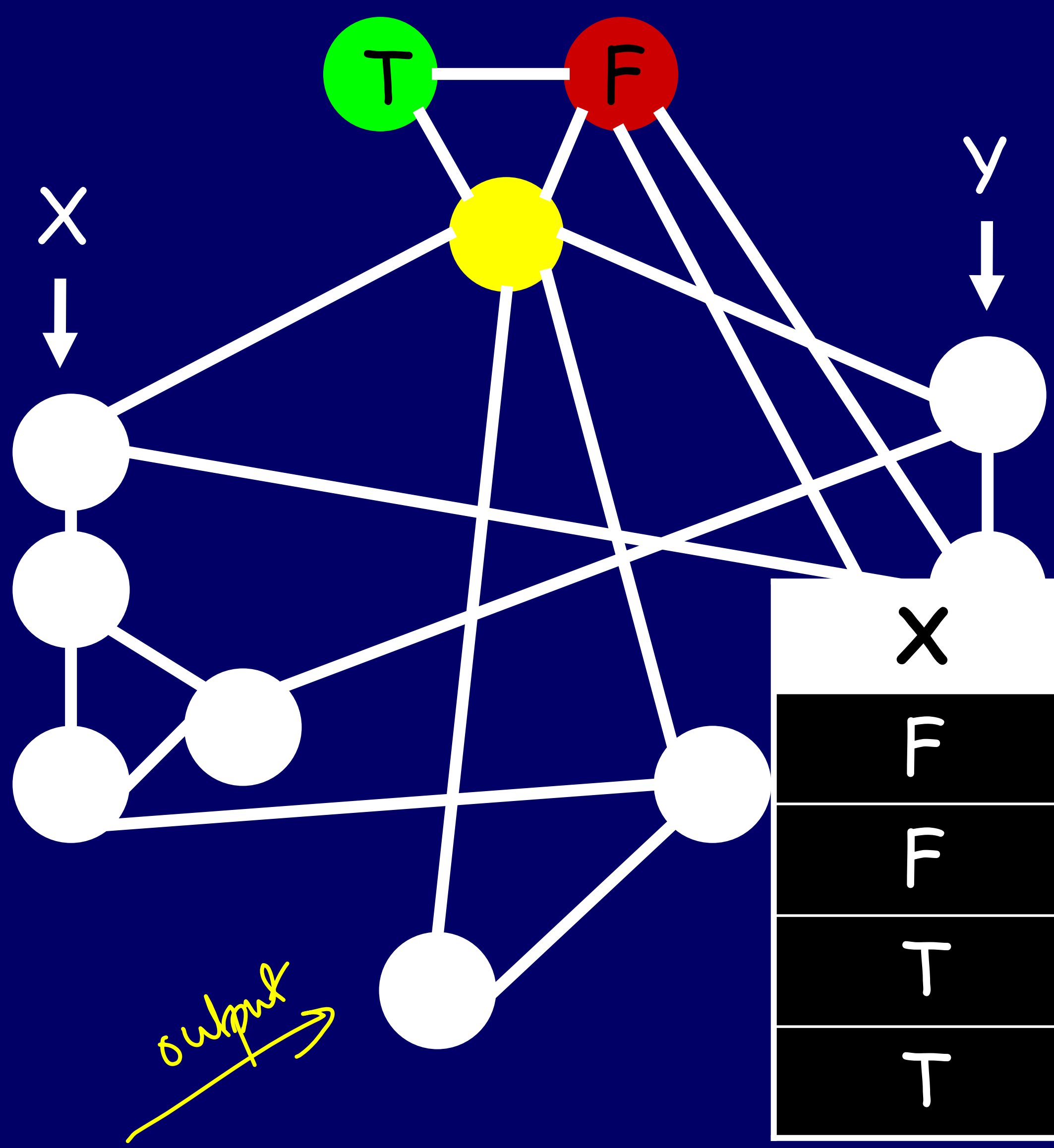




T
F
x
y
G
T
R
Y
G
R
F
? Y
? G
R
G
T

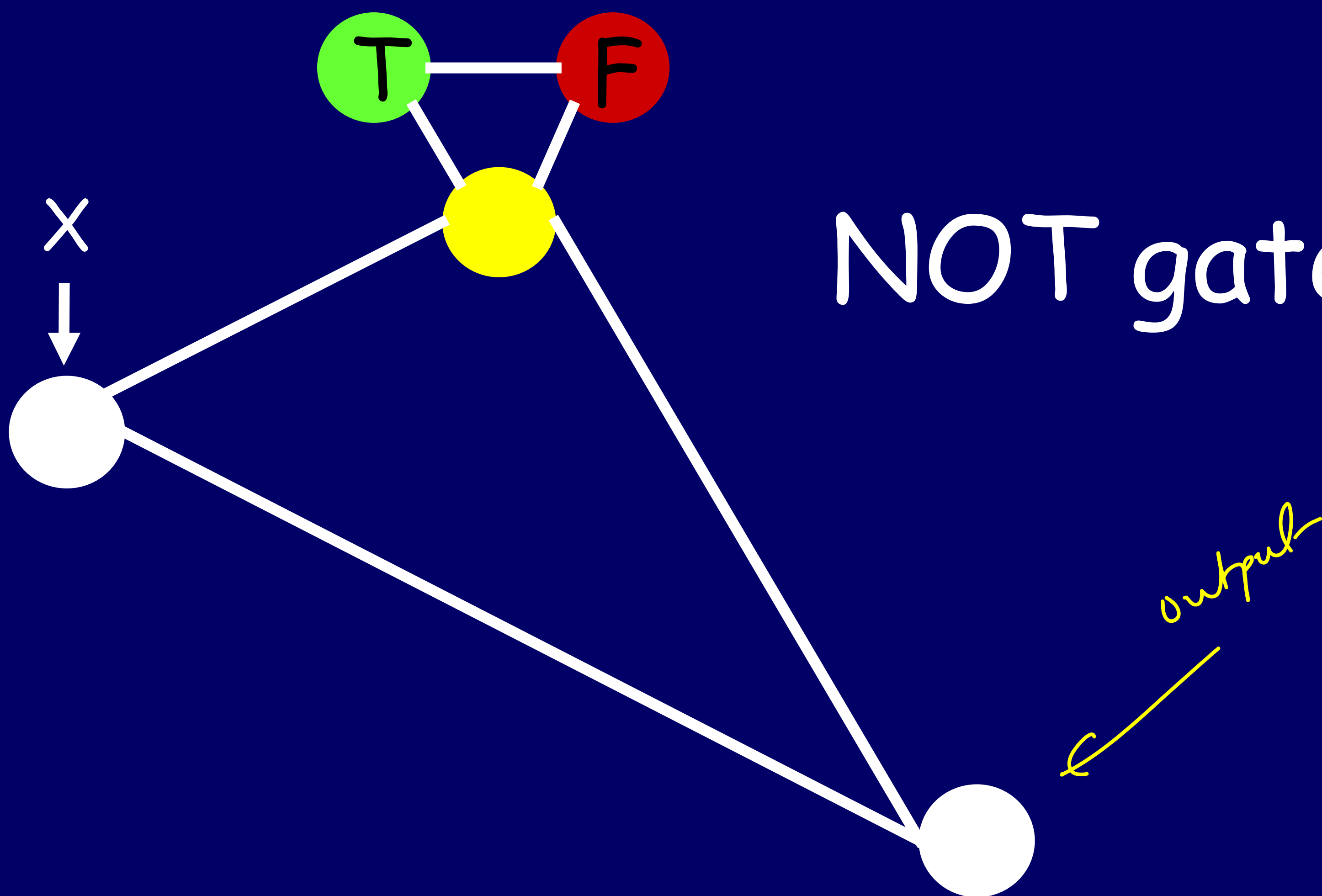


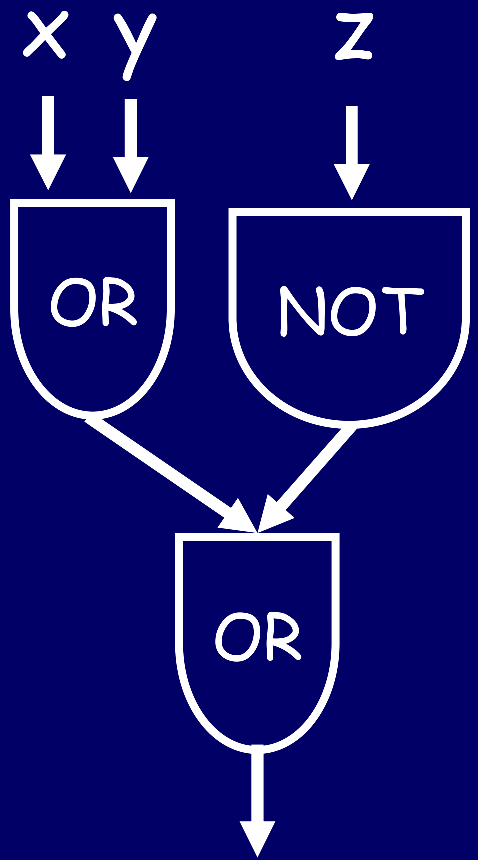
T
F
x
y
R
R

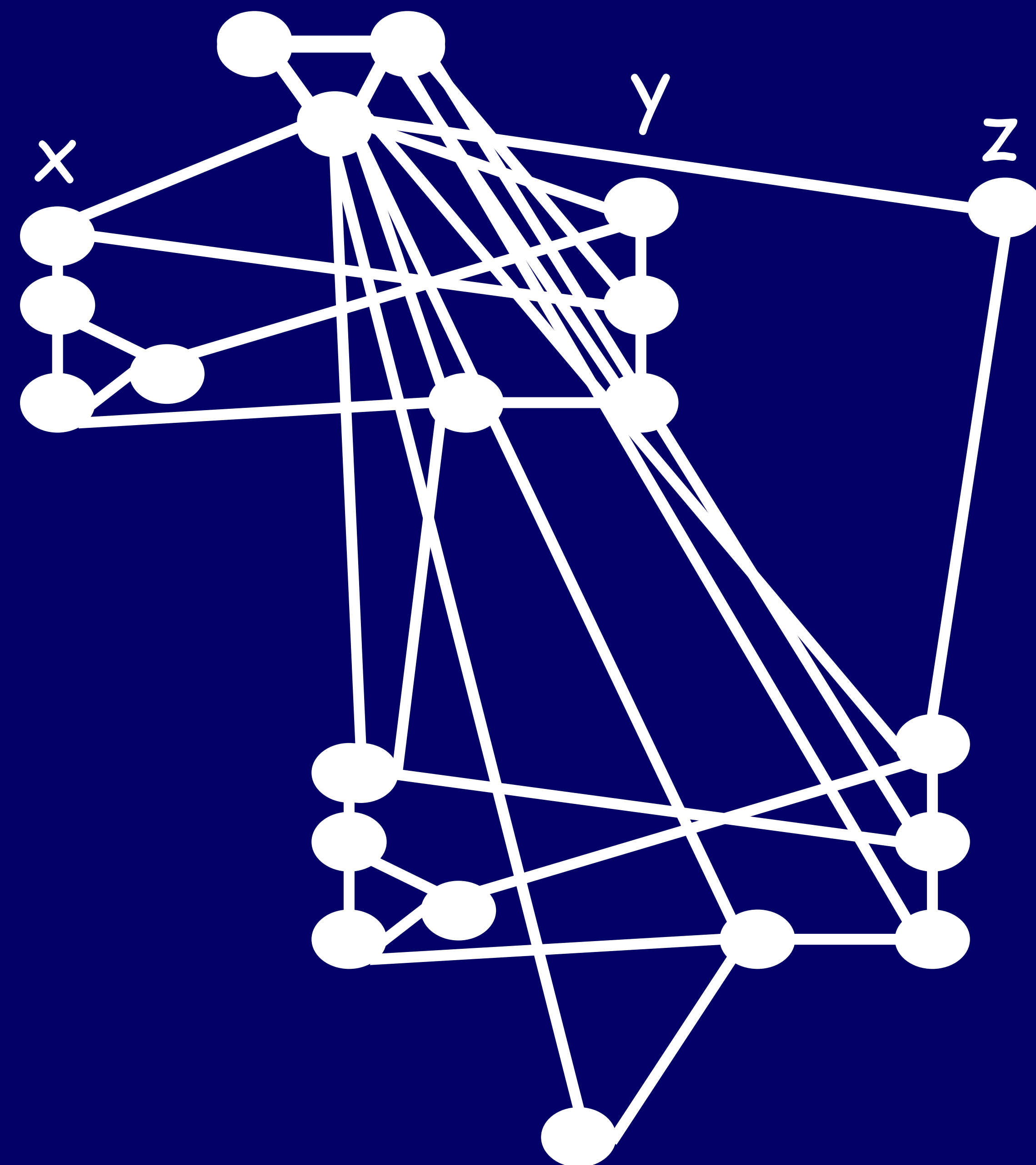


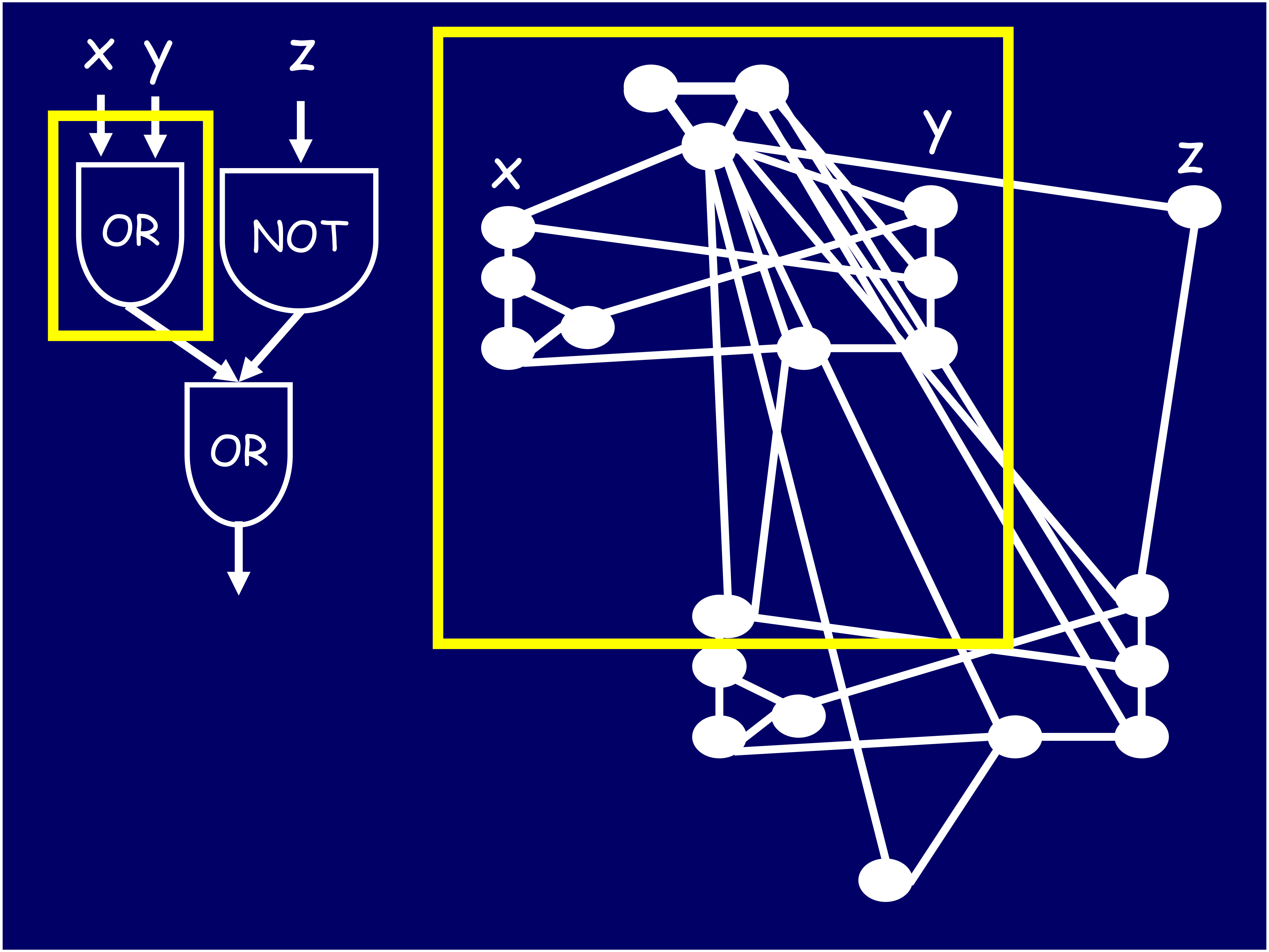
x	y	OR
F	F	F
F	T	T
T	F	T
T	T	T

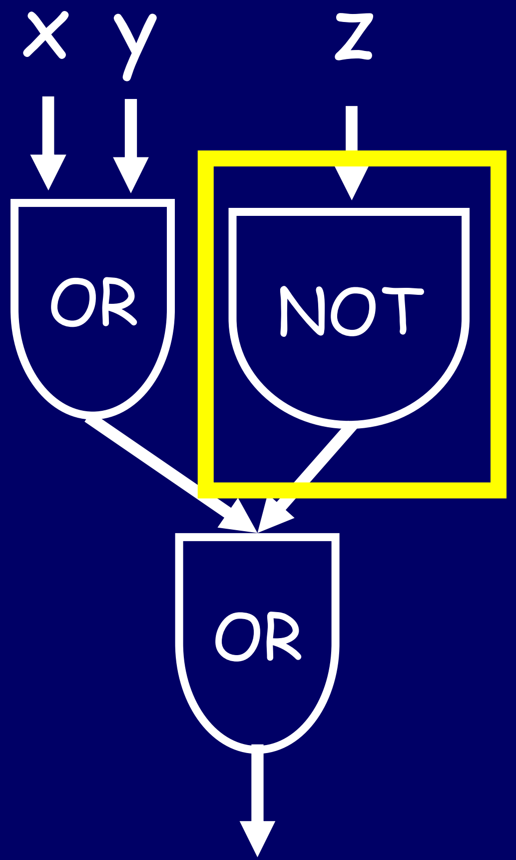
NOT gate!



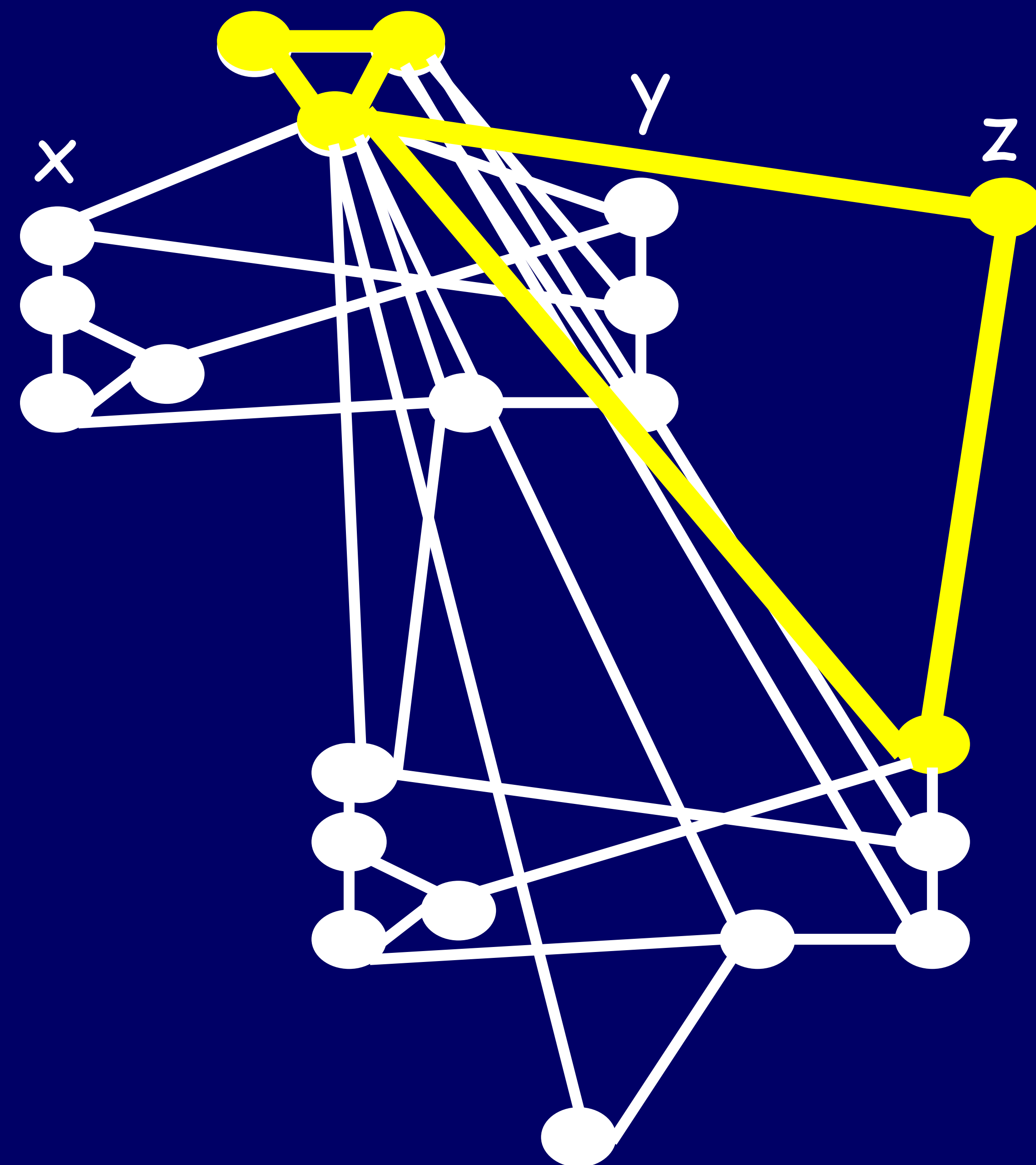

x
y
z
OR
NOT
OR


x
y
z

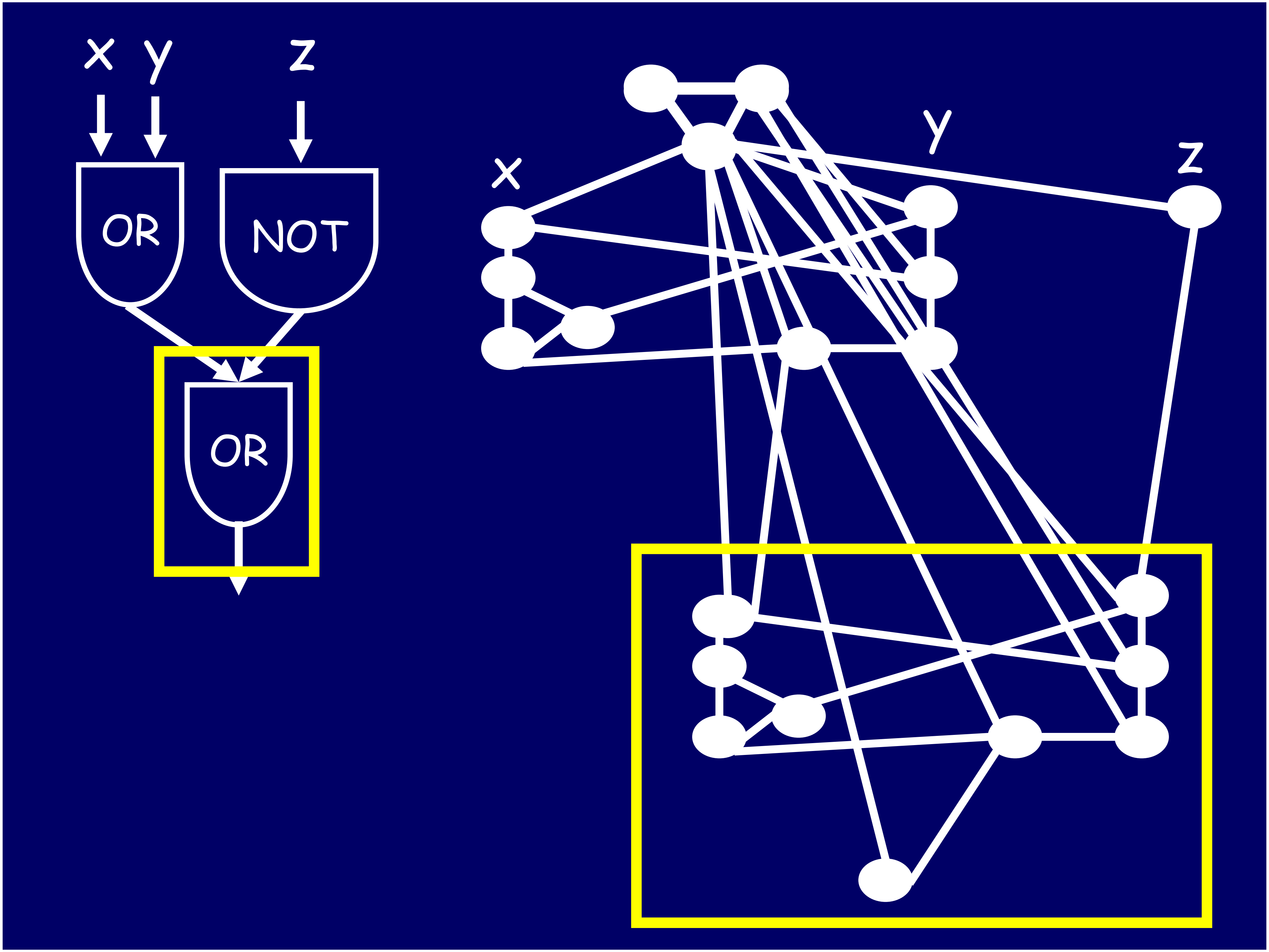

x
y
z
OR
NOT
OR
x
y
z

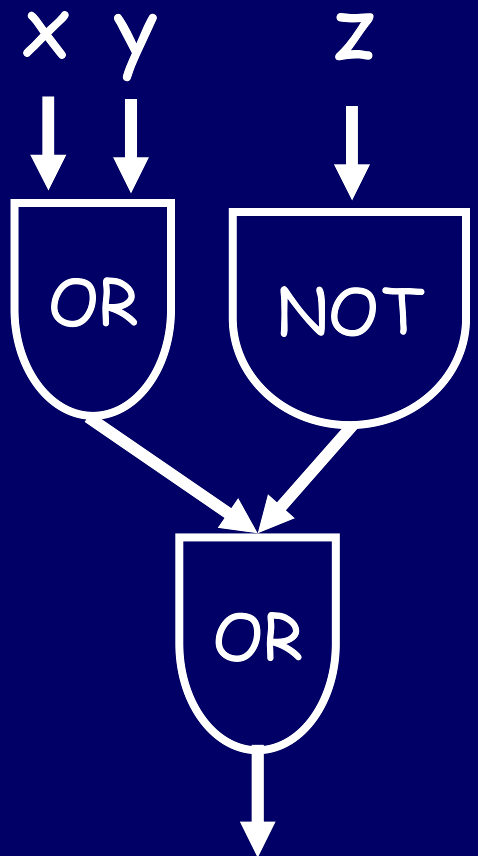


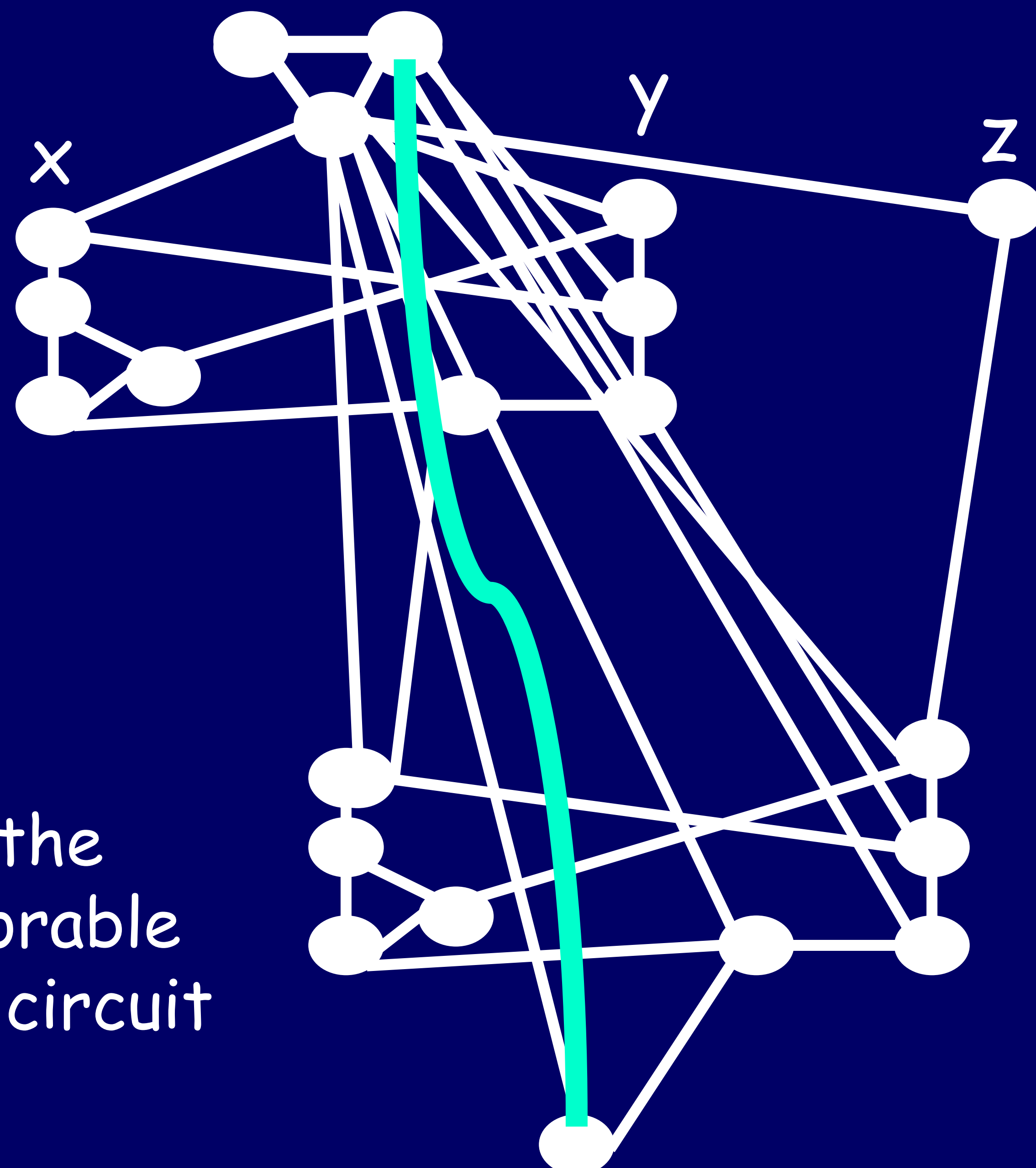
x
y
z
OR
NOT
OR



x
y
z


x
y
z
OR
NOT
OR
x
y
z



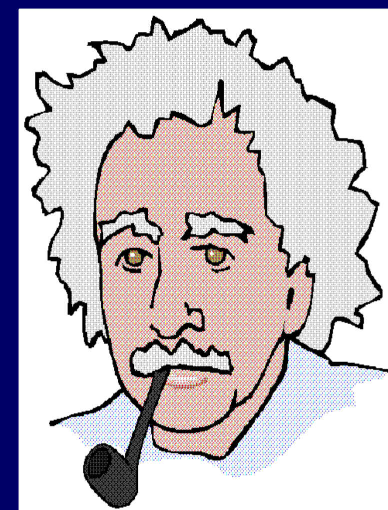


How do we force the graph to be 3 colorable exactly when the circuit is satifiable?

Let C be an n-input circuit

C

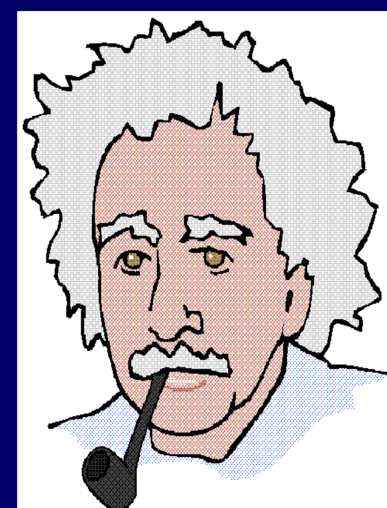
Graph composed of gadgets that mimic the gates in C

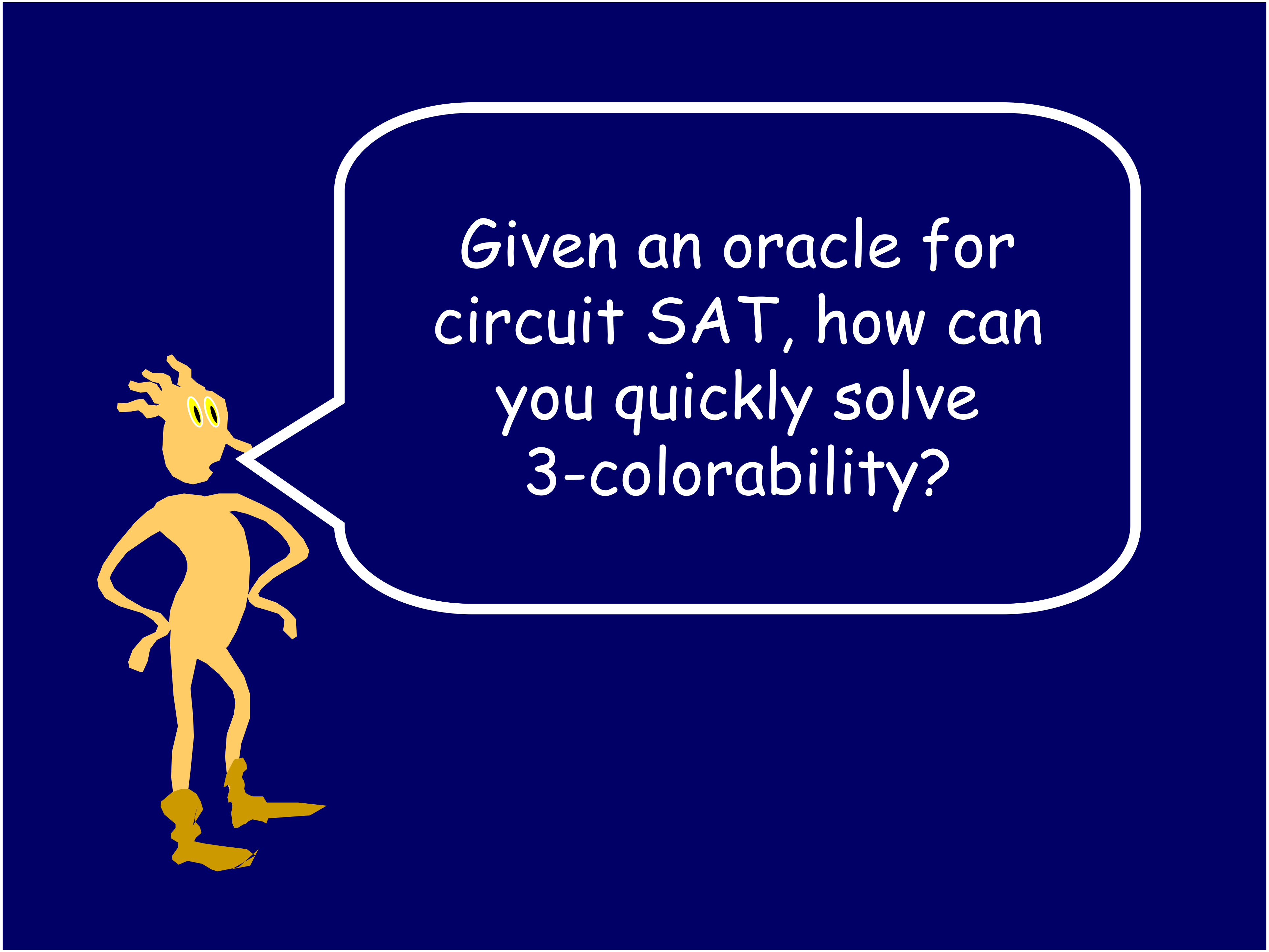


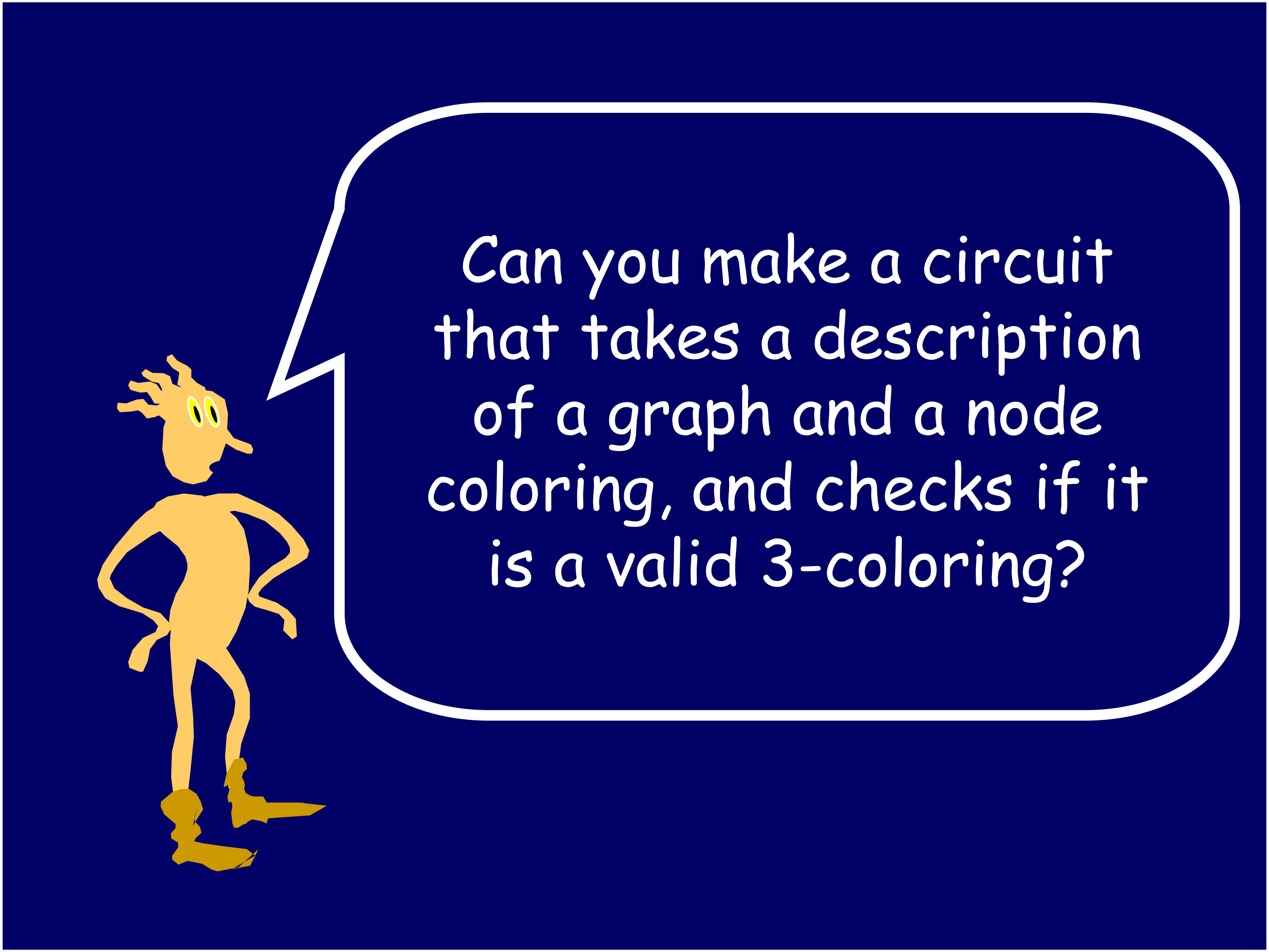
BUILD:
SAT
Oracle

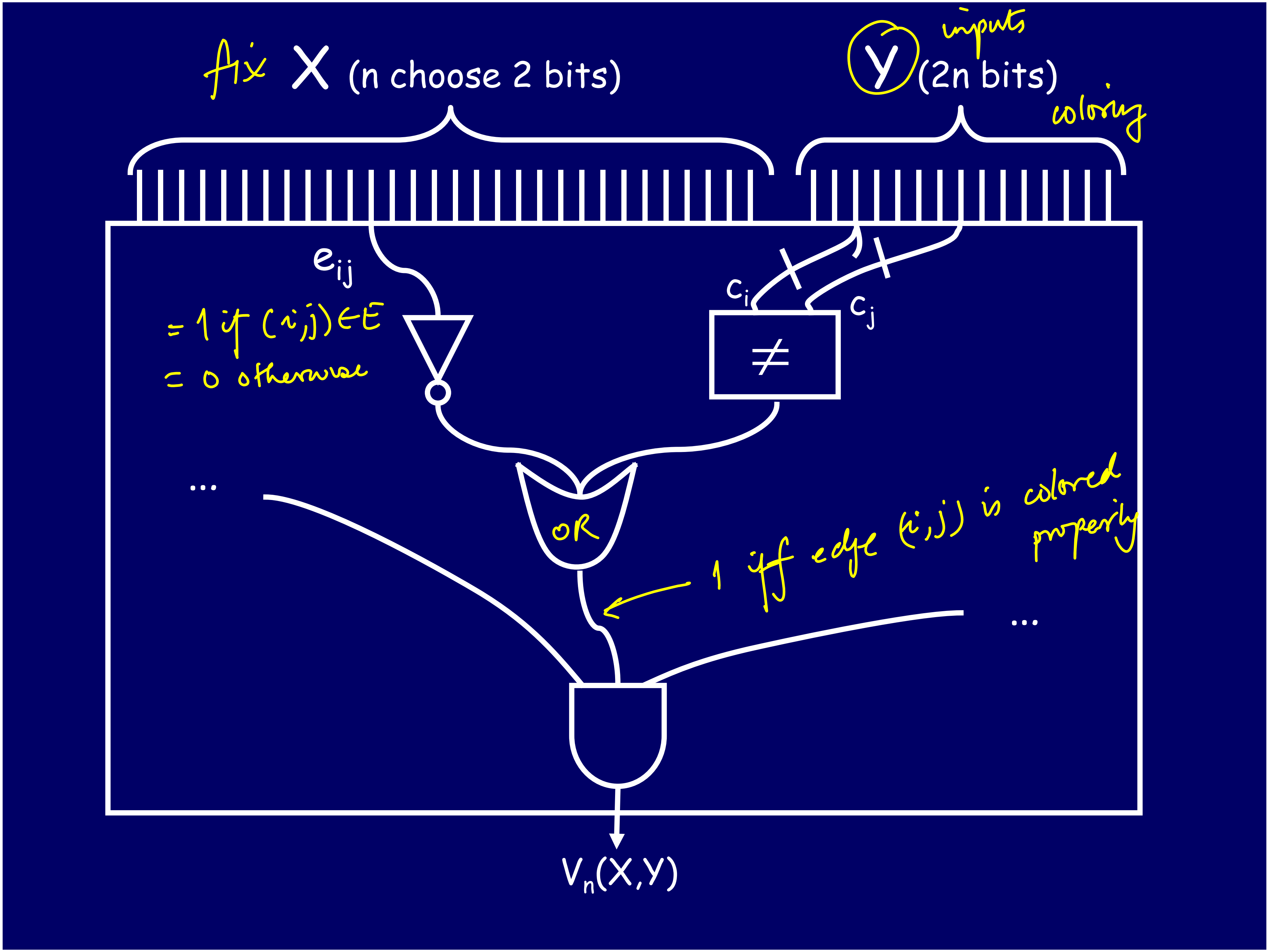
GIVEN:
3-color
Oracle

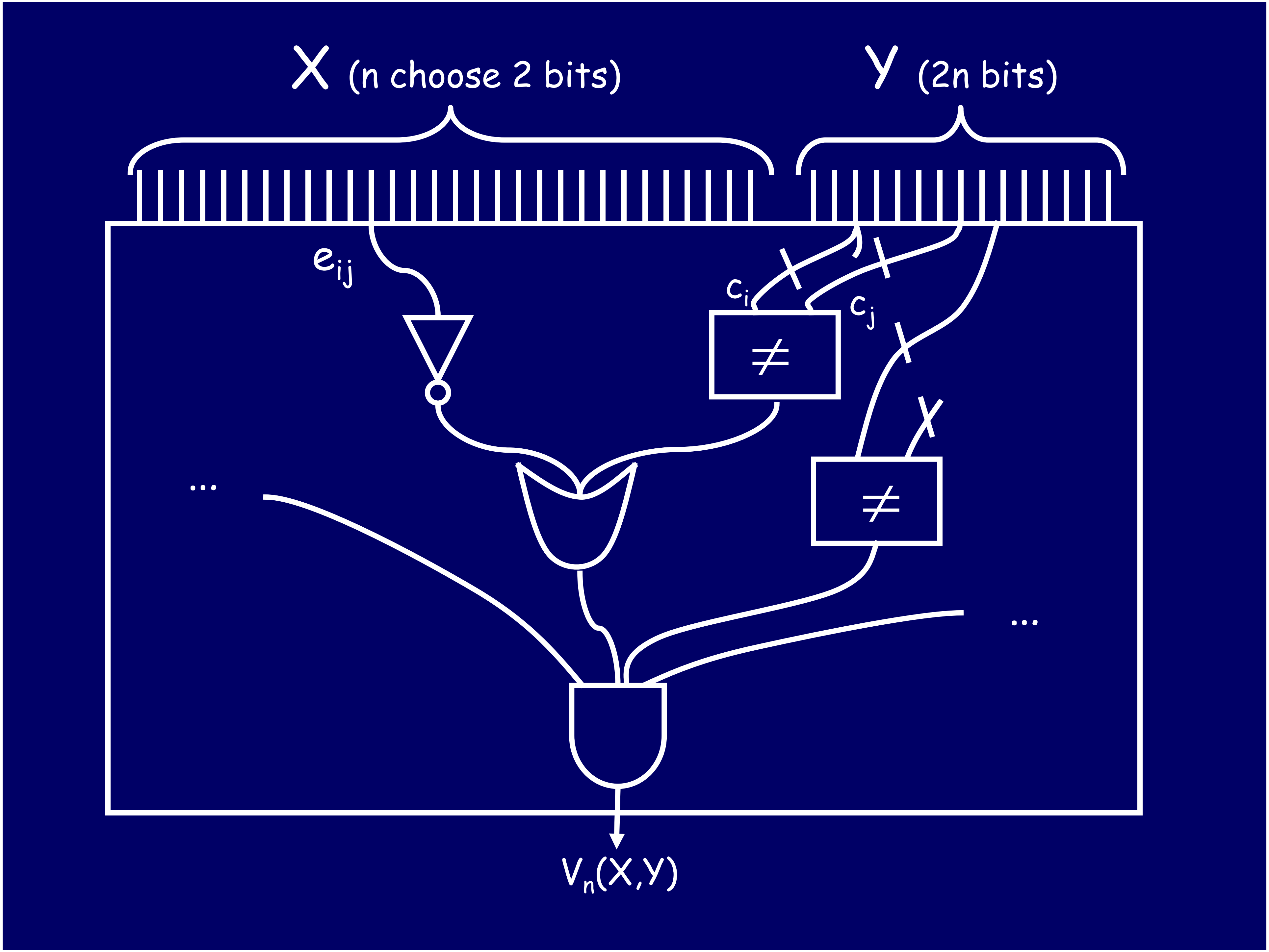
You can quickly transform
a method to decide 3-coloring
into
a method to decide circuit
satifiability!




Given an oracle for circuit SAT, how can you quickly solve 3-colorability?


Can you make a circuit that takes a description of a graph and a node coloring, and checks if it is a valid 3-coloring?


fix X (n choose 2 bits)
Y (2n bits)
inputs
coloring
e_{ij}
= 1 if (i,j) ∈ E
= 0 otherwise
c_i
c_j
≠
OR
1 iff edge (i,j) is colored properly
...
...
$V_n(X,Y)$



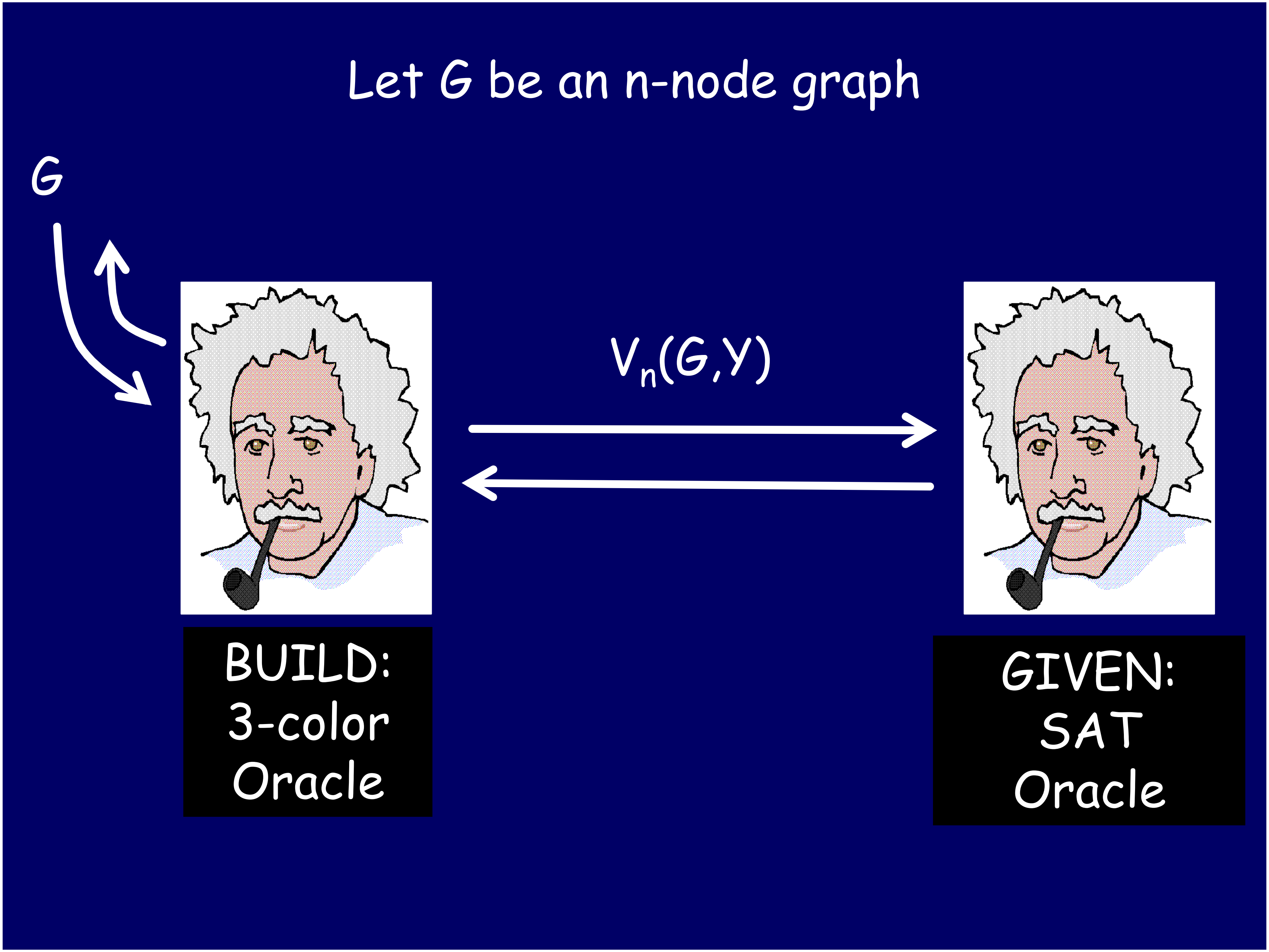
X (n choose 2 bits)
Y (2n bits)
e_{ij}
c_i
c_j
≠
≠
...
...
$V_n(X,Y)$

$V_n(X,Y)$

Let V_n be a circuit that takes an n-node graph X and an assignment of colors to nodes Y, and **verifies** that Y is a valid 3 coloring of X. I.e., $V_n(X,Y) = 1$ iff Y is a 3 coloring of X

X is expressed as an n choose 2 bit sequence. Y is expressed as a 2n bit sequence

Given n, we can construct V_n in time $O(n^2)$

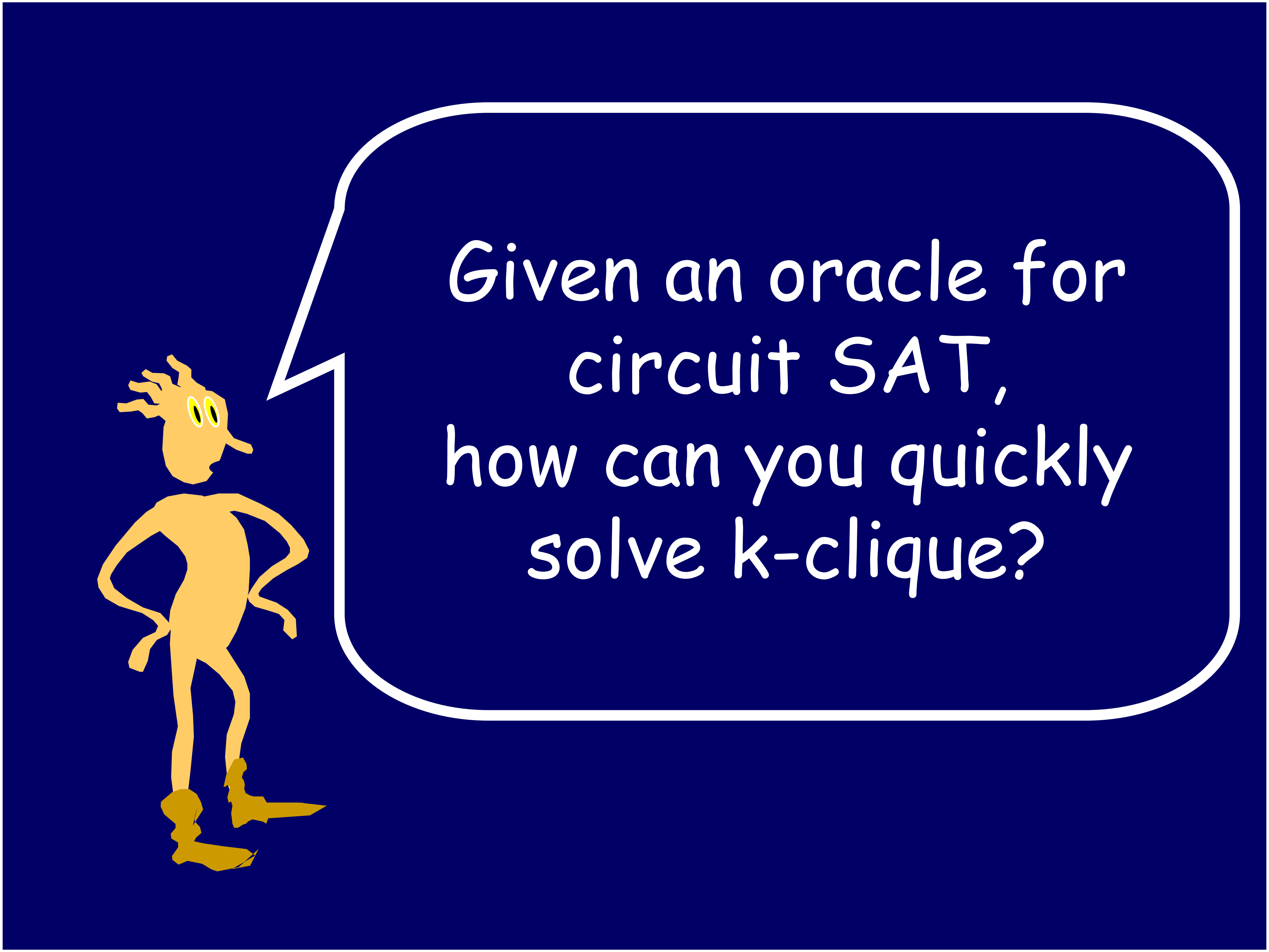

Let G be an n-node graph
G
$V_n(G,Y)$
BUILD:
3-color
Oracle
GIVEN:
SAT
Oracle

Circuit-SAT / 3-Colorability

Two problems that are cosmetically different, but substantially the same

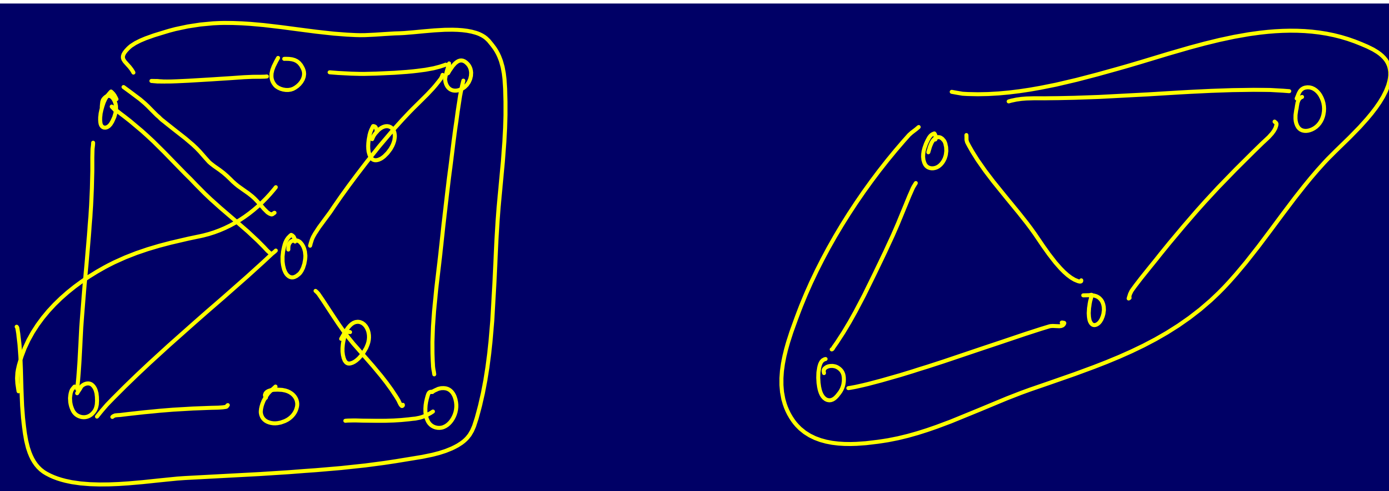
Circuit-SAT / 3-Colorability

Clique / Independent Set


Given an oracle for circuit SAT, how can you quickly solve k-clique?

Circuit-SAT / 3-Colorability

Clique / Independent Set



Four problems that are cosmetically different, but substantially the same

FACT: No one knows a way to solve any of the 4 problems that is fast on all instances

Summary

Many problems that appear different on the surface can be efficiently reduced to each other, revealing a deeper similarity