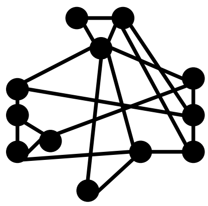
Algorithm Design and Analysis

Victor Adamchik — CS 15-451 — Spring 2014

Lecture 32 — Apr 07, 2014 — Carnegie Mellon University

Efficient Reductions



Plan

Search vs. Decision (3-coloring)

Many-one Reduction (clique/ind. Set)

Circuit-SAT and 3-SAT

Reductions

Comparison between a mathematician and an engineer:

Put an empty kettle in the middle of the kitchen floor and tell your subjects to boil some water.

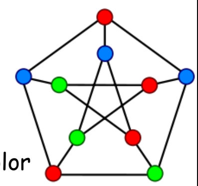
The engineer will fill the kettle with water, put it on the stove, and turn the flame on. The mathematician will do the same thing.

Next, put the kettle already filled with water on the stove, and ask the subjects to boil the water.

The engineer will turn the flame on.

The mathematician will empty the kettle and put it in the middle of the kitchen floor... thereby reducing the problem to one that has already been solved!

K-Coloring



We define a k-coloring of a graph:

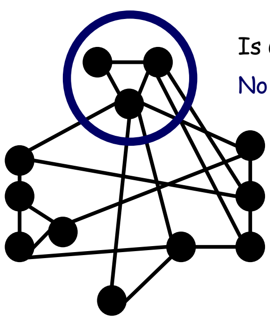
Each node gets colored with one color

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 (aka bipartite)?

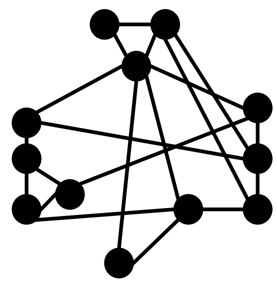
Theorem: G contains an odd cycle if and only if G is not 2-colorable

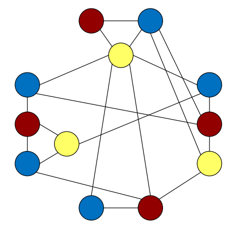
Efficient 2-coloring algorithm:

- 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 graph is not 2-colorable (the conflict proves no 2-coloring is possible, and there is an odd cycle)

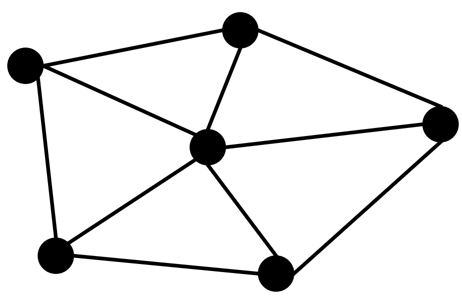
A 3-CRAYOLA Question!

Is this graph 3-colorable?





A 3-CRAYOLA Question!

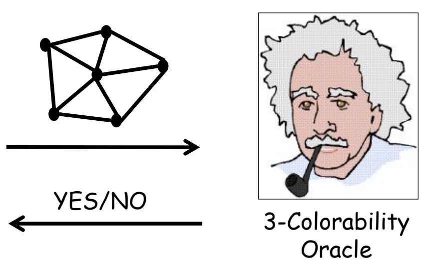


Is the "wheel" 3-colorable?

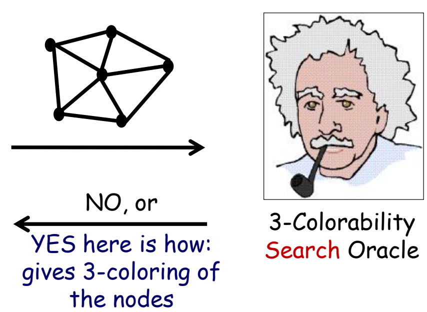
3-Coloring Is Decidable by Brute Force

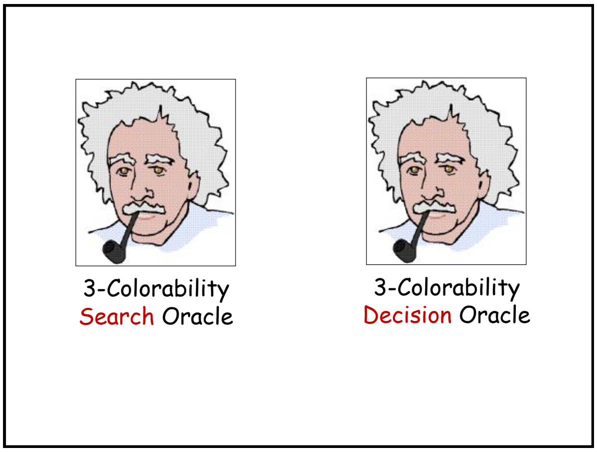
Try out all 3^n colorings until you determine if G has a 3-coloring

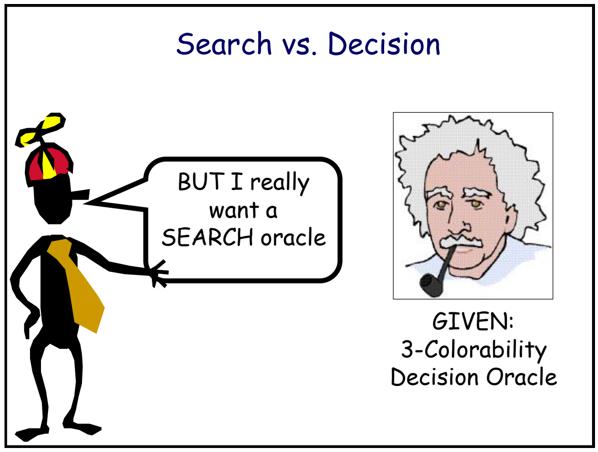
A 3-CRAYOLA Oracle

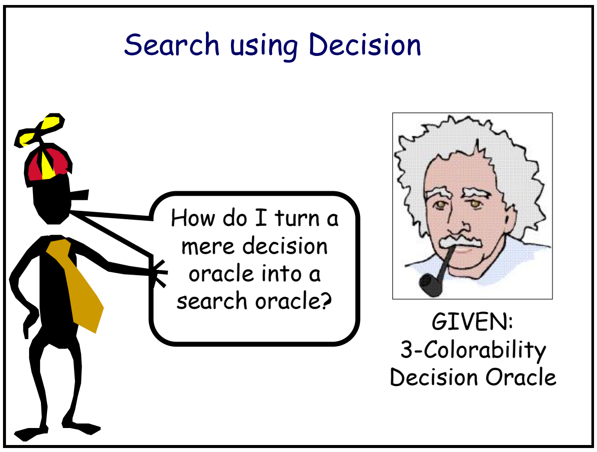


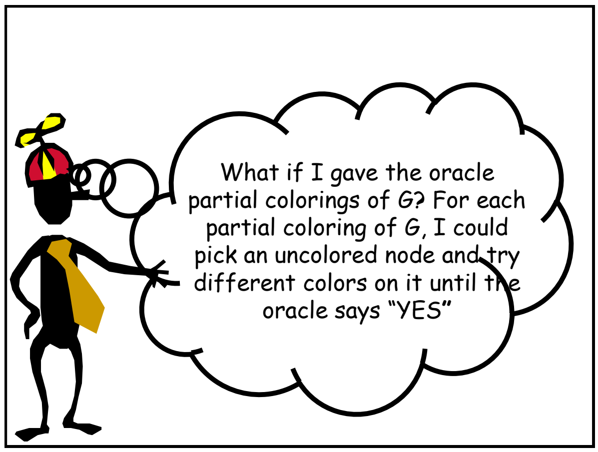
Better 3-CRAYOLA Oracle

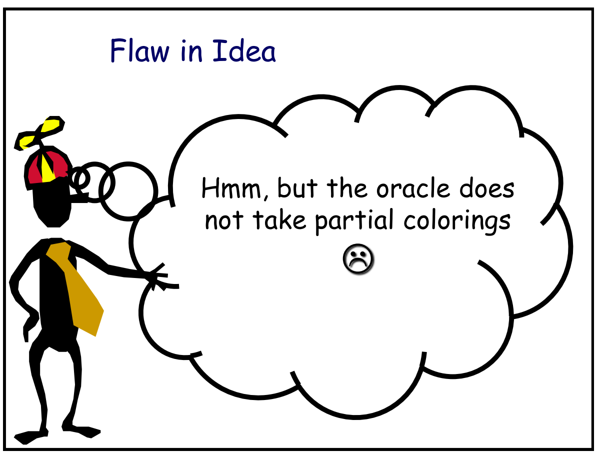


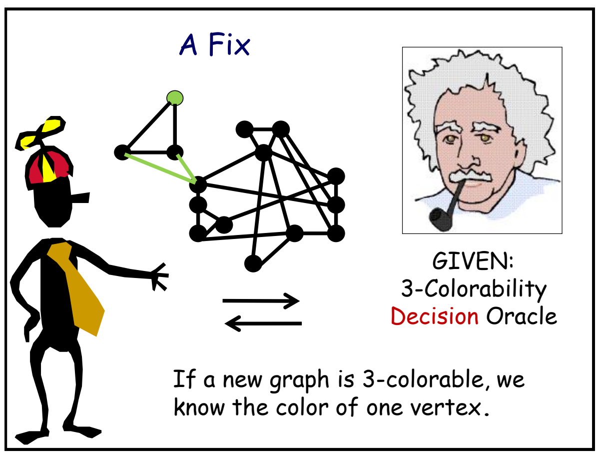

3-Colorability
Search Oracle
3-Colorability
Decision Oracle


Search vs. Decision
BUT I really want a SEARCH oracle
GIVEN:
3-Colorability
Decision Oracle

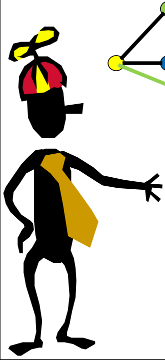

Search using Decision
How do I turn a mere decision oracle into a search oracle?
GIVEN:
3-Colorability
Decision Oracle

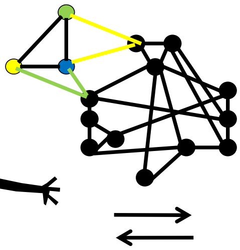

What if I gave the oracle partial colorings of G? For each partial coloring of G, I could pick an uncolored node and try different colors on it until the oracle says "YES"


Flaw in Idea
Hmm, but the oracle does not take partial colorings ☹


A Fix
GIVEN:
3-Colorability
Decision Oracle
If a new graph is 3-colorable, we know the color of one vertex.

A Fix

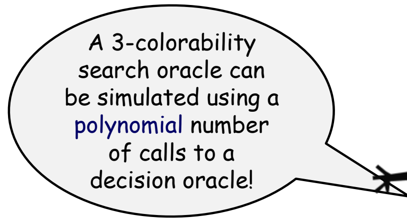




GIVEN:
3-Colorability
Decision Oracle

If a new graph is not 3-colorable, we try another color.

3-colorability oracles



But how efficient the decision oracle?

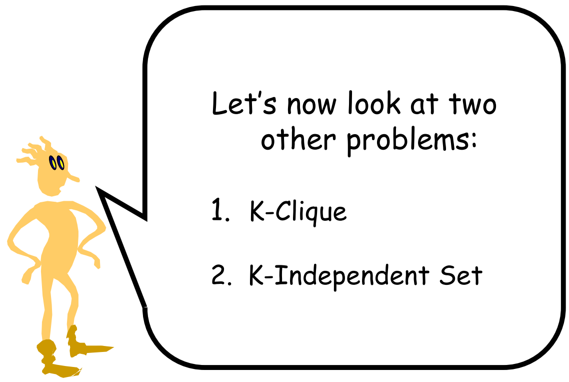
Search vs. Decision

Solving search problem efficiently means that decision problem can b e solved efficiently

...just run the algorithm for the search problem

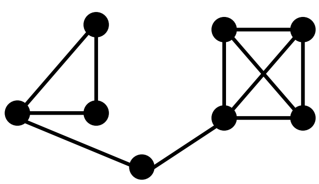
However, if decision problem is difficult then search version is definitely difficult

In some case we can use an algorithm for decision problem to solve the search problem.



K-Cliques

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



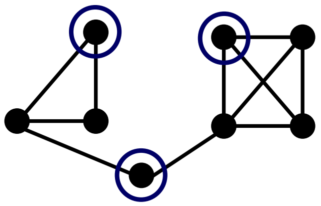
This graph contains a 4-clique

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

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

---

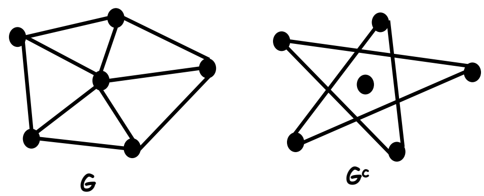
Clique / Independent Set

Two problems that are cosmetically different, but substantially the same

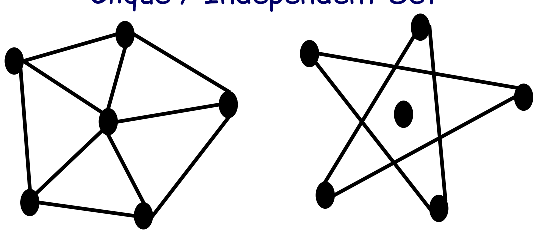
---

Complement of G

Given a graph G, let G^c, the complement of G, be the graph obtained by the rule that two nodes in G^c are connected if and only if the corresponding nodes of G are not connected

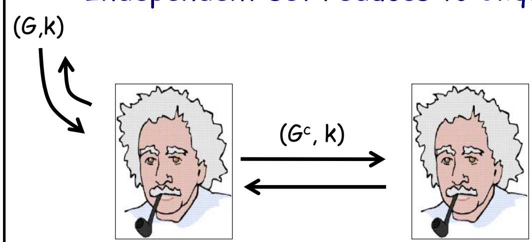


Clique / Independent Set



G has a k-clique ⇔ G^c has an independent set of size k

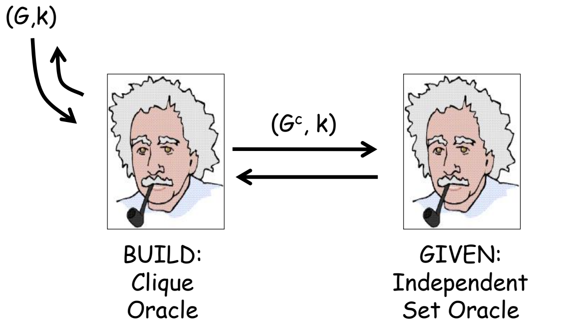
Independent set reduces to Clique

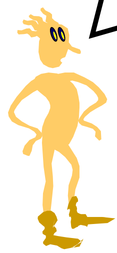


BUILD: Independent Set Oracle

GIVEN: Clique Oracle

Clique reduces to Independent set





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

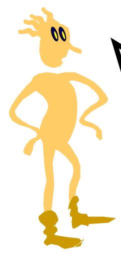
There is a fast method for one if and only if there is a fast method for the other

Many-one reduction

To reduce problem A to problem B (we write $A \leq_p B$) we want a function f that maps A to B such that:

1) f is a polynomial time computable
2) $x \in A$ if and only if $f(x) \in B$.

This also called Karp's reduction and mapping reduction

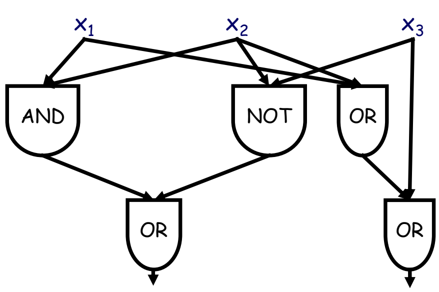


Let's now look at two other problems:

1. Circuit Satisfiability
2. Graph 3-Colorability

Combinatorial Circuits

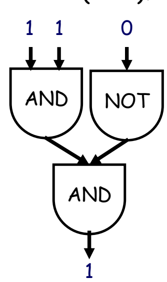
AND, OR, NOT, 0, 1 gates wired together with no feedback allowed



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

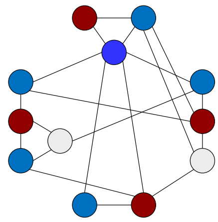


Circuit-Satisfiability

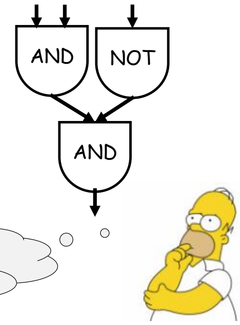
Given: A circuit with n-inputs and one output, is there a way to assign 0-1 values to the input s so that the output value is 1 (true)?

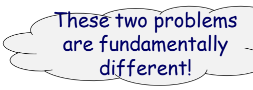
BRUTE FORCE: Try out all 2^n assignments

3-Colorability



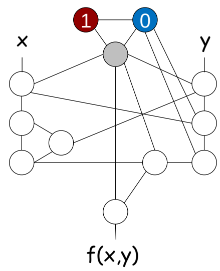
Circuit Satisfiability





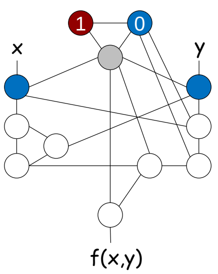
Given an oracle for 3-colorability, how can you quickly solve circuit SAT?

3-colorability vs. circuit-Sat



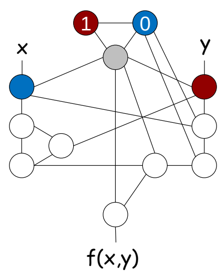
x	y	$f(x,y)$

3-colorability vs. circuit-Sat



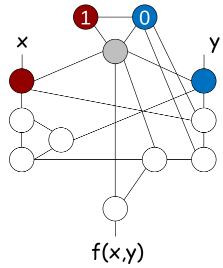
x	y	$f(x,y)$
0	0	0

3-colorability vs. circuit-Sat



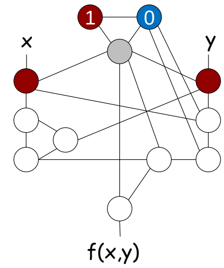
x	y	$f(x,y)$
0	0	0
0	1	1

3-colorability vs. circuit-Sat



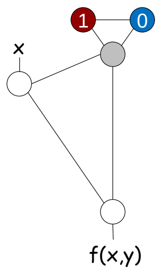
x	y	$f(x,y)$
0	0	0
0	1	1
1	0	1

3-colorability vs. circuit-Sat



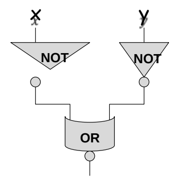
x	y	OR
0	0	0
0	1	1
1	0	1
1	1	1

3-colorability vs. circuit-Sat



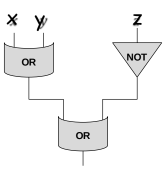
x	NOT
0	1
1	0

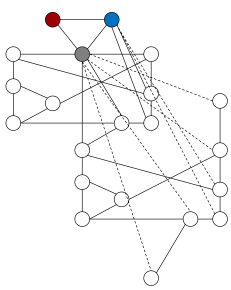
3-colorability vs. circuit-Sat



AND Gate from OR and NOT

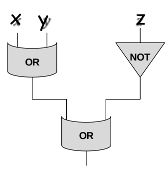
3-colorability vs. circuit-Sat

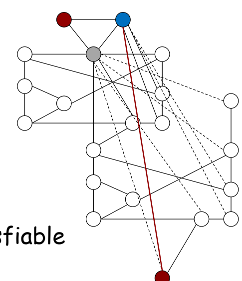




How do we force the graph to be 3 colorable exactly when the circuit is satifiable?

3-colorability vs. circuit-Sat



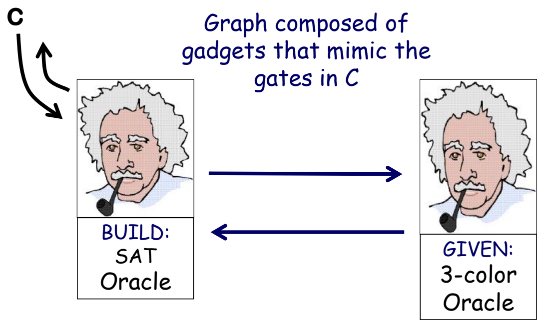


Circuit is satisfiable

⇕

Graph is 3-colorable

Let C be an n-input circuit.



3-colorability vs. circuit-Sat

There is a linear-time function that reduces instances of CIRCUIT-SAT to instances of 3-COLORABILITY

CIRCUIT-SAT $\leq_p$ 3-COLOR

Fact: There are efficient ways to reduce an instance of any of the four problems we discussed to an instance of any other