

Magic Trick

MAP OF TETRISTAN

8 countries

Color the map - no two touching countries can have the same color

3 colors, # colors {0, 1, 2}

$$D + F + C \times (G - A) + H = 3$$

Quiz: What's a good way to find an upper bound on a problem's hardness?

- a Devise an algorithm to solve it. Run it on a bunch of inputs. The shape of the graph is the bound.
- a Devise an algorithm to solve it. Run on all possible inputs - running time is the upper bound.
- a Devise an algorithm to solve it. Analyze it. Big O/Θ is UB.
- a No idea - is it studied in some other class?

## Lower Bound on Complexity

Finding the max of a list ( $n$ )

Upper bound:  $\Theta(n)$

lower bound?

- $\Theta(1)$ : need to give the answer. Maybe guessable...
- $\Theta(n)$ : must at least look at all  $n$  items (or could miss the max)
- $\Theta(\log n)$ : like a tournament, might be able to eliminate half each time.

## Longest Simple Path

Long & Simple Path

Simple Path:  
no repeated nodes

a-g

any shortest path - simple  
longest simple path -  $n$

check box K if there is  
a simple path of  $K$  nodes from a to g

1 ☐  
2 ☐  
3 ☐  
4 ☒  
5 ☒  
6 ☒  
7 ☒

## Reduction: Long and Simple Path

```
def long_and_simple_path(G,u,v,l):
    """
    G: Graph
    u: starting node
    v: ending node
    l: minimum length of path
    """
    if not long_and_simple_decision(G,u,v,l):
        return False
    for node1 in G:
        neighbors = G[node1].keys()
        for node2 in neighbors:
            G = break_link(G, node1, node2)
            if not long_and_simple_decision(G,u,v,l):
                G = make_link(G, node1, node2)
    path = [u]
    node = u
    next_node = (G[node].keys())[0]
    while next_node != v:
        path.append(next_node)
        next_next_0 = (G[next_node].keys())[0]
        next_next_1 = (G[next_node].keys())[1]
        if next_next_0 == node:
            node, next_node = next_node, next_next_1
        else:
            node, next_node = next_node, next_next_0
    return path + [v]
```

## Accepting Certificate

$P \subseteq NP?$

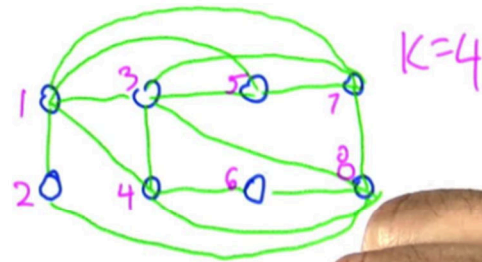
- o IF I knew that, I'd win a Fields Medal!
- o No, if it is decidable in Polynomial time, no certificate is needed.
- o Yes, if it can be decided in Polynomial time, no certificate is needed.

## Clique Problem

### Clique Problem

Given graph  $G$  and number  $k$ , is there a clique of size  $k$  in  $G$ ?

16



$P=NP?$

What if  $P=NP$ ?

- Some cryptographic protocols based on problems like factoring could be cracked.
- A lot of CS theoreticians will be out of work.
- Computers will be smarter than people

## Find the Strangers

$s$ -independent set Problem Find the strangers!

Given graph  $H$  and number  $s$  is there a set of nodes of size  $s$  in  $H$  such that no two nodes in the set are connected in  $H$ ? (they are independent of each other)

Reduce to  $k$ -clique: Show how a polytime solution to  $k$ -clique solves

- $s$ -independent can be solved by guessing a set of nodes and <sup>it</sup> seeing if no pair is connected. Check  $s^2$  edges, so it is poly time.
- Run  $s$ -clique algorithm on  $H$ . Return the opposite of the answer.
- Let  $G$  be the complement of  $H$ . Run  $s$ -clique on  $G$ , return opposite.
- Let  $G$  be the complement of  $H$ . Run  $s$ -clique on  $G$ , return answer.

## NP-Completeness

Graph Partitioning is NP complete (GP)

Even if you don't know what it is,  
check all you know

- ☑ GP is NP-hard - a polytime solution solves everything in NP
- ☑ clique can be reduced to it
- ☑ GP is in NP - an exponential-time solution exists
- ☑ It can be reduced to SAT.



### Solving 3-Colorability

```
def verify(G, cert, k):  
    if len(cert) != len(G):  
        return False  
    for node in cert:  
        if cert[node] not in range(k):  
            return False  
        for neighbor in G[node]:  
            if cert[node] == cert[neighbor]:  
                return False  
    return True
```

### Generating a Formula

Formula Size

K colors 3  
N nodes 8  
M edges 20

How many clauses in the Formula?  
(anded together)

92

## 4 colorability Quiz

- ☒ It's in NP because we can quickly verify a certificate that lists the color assignments
- ☐ It's not necessarily in NP because the number of colors is bigger
- ☐ It's not necessarily NP hard because having 4 colors to work with makes things easier
- ☒ It's NP hard because a solution to 4 colorability solves 3 colorability - add a node & connect it to all
- ☐ Since any graph that is 3 colorable is 4 colorable - equivalent