Is the graph $G = (V, E)$ $k$-colorable?

Is there a $c : V \rightarrow \{1, 2, \ldots, k\}$ such that $c(u) \neq c(v)$ for $uv \in E$?
Is the graph $G = (V, E)$ $k$-colorable?

Is there a $c : V \rightarrow \{1, 2, \ldots, k\}$ such that $c(u) \neq c(v)$ for $uv \in E$?
Is the graph $G = (V, E)$ $k$-colorable?

Is there a $c : V \rightarrow \{1, 2, \ldots, k\}$ such that $c(u) \neq c(v)$ for $uv \in E$?
Largest clique lower bounds the number of colors needed
Largest clique lower bounds the number of colors needed
Largest clique lower bounds the number of colors needed
Largest clique lower bounds the number of colors needed
Graph coloring is a classical problem

A central problem with many applications:
- Frequency assignment
- Scheduling
- Register allocation
- Puzzles (e.g., Sudoku)
- ...
Finding an optimal coloring is hard
Finding an optimal coloring is hard
Finding an optimal coloring is hard
Finding an optimal coloring is hard
Finding an optimal coloring is hard
Many ways to fight hard problems

- Faster exponential-time algorithms
Many ways to fight hard problems

- Faster exponential-time algorithms
- Exploit structural properties
Many ways to fight hard problems

- Faster exponential-time algorithms
- Exploit structural properties
- Probabilistic algorithms
Many ways to fight hard problems

- Faster exponential-time algorithms
- Exploit structural properties
- Probabilistic algorithms
- Approximation algorithms
Many ways to fight hard problems

- Faster exponential-time algorithms
- Exploit structural properties
- Probabilistic algorithms
- Approximation algorithms
- Heuristic methods
Improve expected runtime with a SAT solver

Variables take values from \{0, 1\}

Clauses are a disjunction of variables

Example: \((x_1 \lor x_2) \land (x_1 \lor \neg x_3)\) — is there a satisfying assignment?

The satisfiability problem (SAT)

There are \(2^n\) assignments to \(n\) variables
Improve expected runtime with a SAT solver

The satisfiability problem (SAT)

- Variables take values from \( \{0, 1\} \)
- Clauses are a disjunction of variables
- Example: \( (x_1 \lor x_2) \land (x_1 \lor \neg x_3) \) — is there a satisfying assignment?
Improve expected runtime with a SAT solver

**The satisfiability problem (SAT)**
- Variables take values from \( \{0, 1\} \)
- Clauses are a disjunction of variables
- Example: \((x_1 \lor x_2) \land (x_1 \lor \neg x_3)\) — is there a satisfying assignment?

**There are \(2^n\) assignments to \( n\) variables**

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(x_3)</th>
<th>( ((x_1 \lor x_2) \land (x_1 \lor \neg x_3)) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 1 1 1 1 1 0 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1 1 1 1 1 1 0 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1 1 0 1 1 1 0 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1 1 0 1 1 1 1 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0 1 1 0 0 0 0 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0 1 1 1 0 1 1 0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0 0 0 0 0 0 0 1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 0 0 0 0 1 1 0</td>
</tr>
</tbody>
</table>
SAT is NP-complete, but has “good” solvers

- Steady SAT solver research already from the 60s
SAT is NP-complete, but has “good” solvers

- Steady SAT solver research already from the 60s
- Faster algorithms, better data structures, smarter heuristics, ...
SAT is NP-complete, but has “good” solvers

- Steady SAT solver research already from the 60s
- Faster algorithms, better data structures, smarter heuristics, ...
- A modern solver routinely handles instances thought to be out of reach a decade ago
SAT is NP-complete, but has “good” solvers

- Steady SAT solver research already from the 60s
- Faster algorithms, better data structures, smarter heuristics, ...
- A modern solver routinely handles instances thought to be out of reach a decade ago
- Still NP-complete — small structured instances very difficult for any solver
The assignment: model graph coloring as SAT

The assignment:
- You are given four graphs \( A, B, C, \) and \( D \)
- “What is the minimum \( k \) for which \( G \) is \( k \)-colorable?”
The assignment: model graph coloring as SAT

The assignment:
- You are given four graphs $A$, $B$, $C$, and $D$
- “What is the minimum $k$ for which $G$ is $k$-colorable?”

Our approach:
- Write a program that takes a graph $G$, and outputs a formula $\varphi$
- It must hold that $G$ is $k$-colorable if and only if $\varphi$ is satisfiable
- Feed $\varphi$ to a solver, and inspect the results
Come up with an encoding, and pick a solver

How do we model graph coloring as SAT?

- A high-level idea provided — work out the details
Come up with an encoding, and pick a solver

How do we model graph coloring as SAT?
  ▶ A high-level idea provided — work out the details

How do we choose a SAT solver?
  ▶ Some solvers recommended — experiment and investigate
Extra challenge: is the graph $D$ 14-colorable?

- An open problem until the early 90s
- You can freely use whatever method to settle the question
- Challenging, but doable (might need some tricks or insight)
Graph coloring through satisfiability

- A famous problem having applications in e.g., telecommunications
- Problem: hard, but we still want a solution somehow
- Solution: model as SAT, feed to a solver, hope for the best

A link to full details from the course homepage. (DL May 8, 2016)