

SAT Solving

Nuno Macedo
(original slides by Alcino Cunha)

Complexity

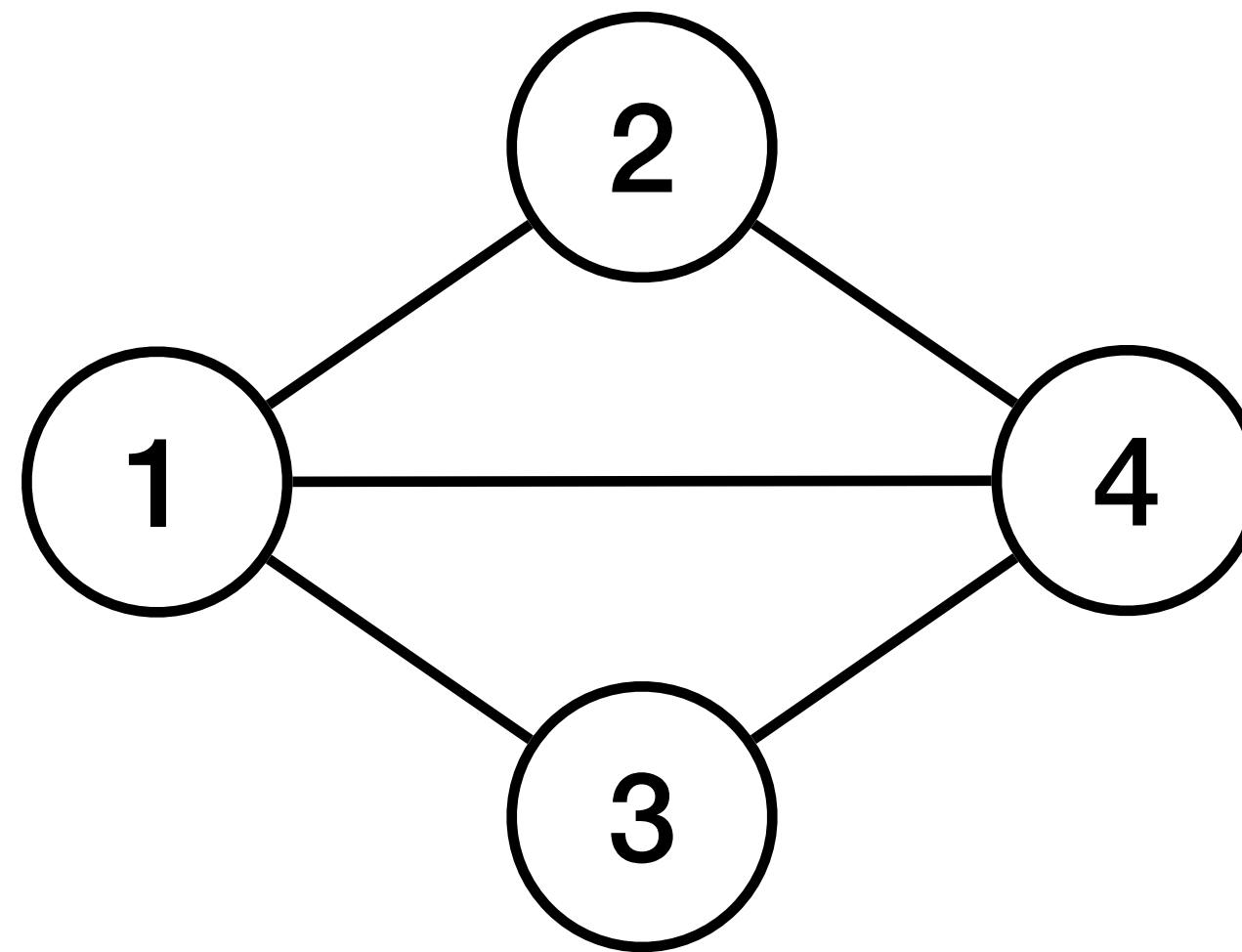
- Given a propositional formula ϕ , the decision problem “Is ϕ satisfiable?” is known as the **Boolean satisfiability problem** or **SAT**
 - SAT is decidable
 - SAT is NP-complete

NP-completeness

- **Nondeterministic polynomial** time (NP) is a complexity class for decision problems
 - Problem instances have “proofs” verifiable in polynomial time
 - SAT is in NP (proofs are assignments)
- A NP problem is **NP-complete** if every problem in NP is reducible to it in polynomial time
 - SAT was the first problem to be show to be NP-complete
 - $P \subseteq NP$ but we do not known if $P = NP$ or $P \neq NP$

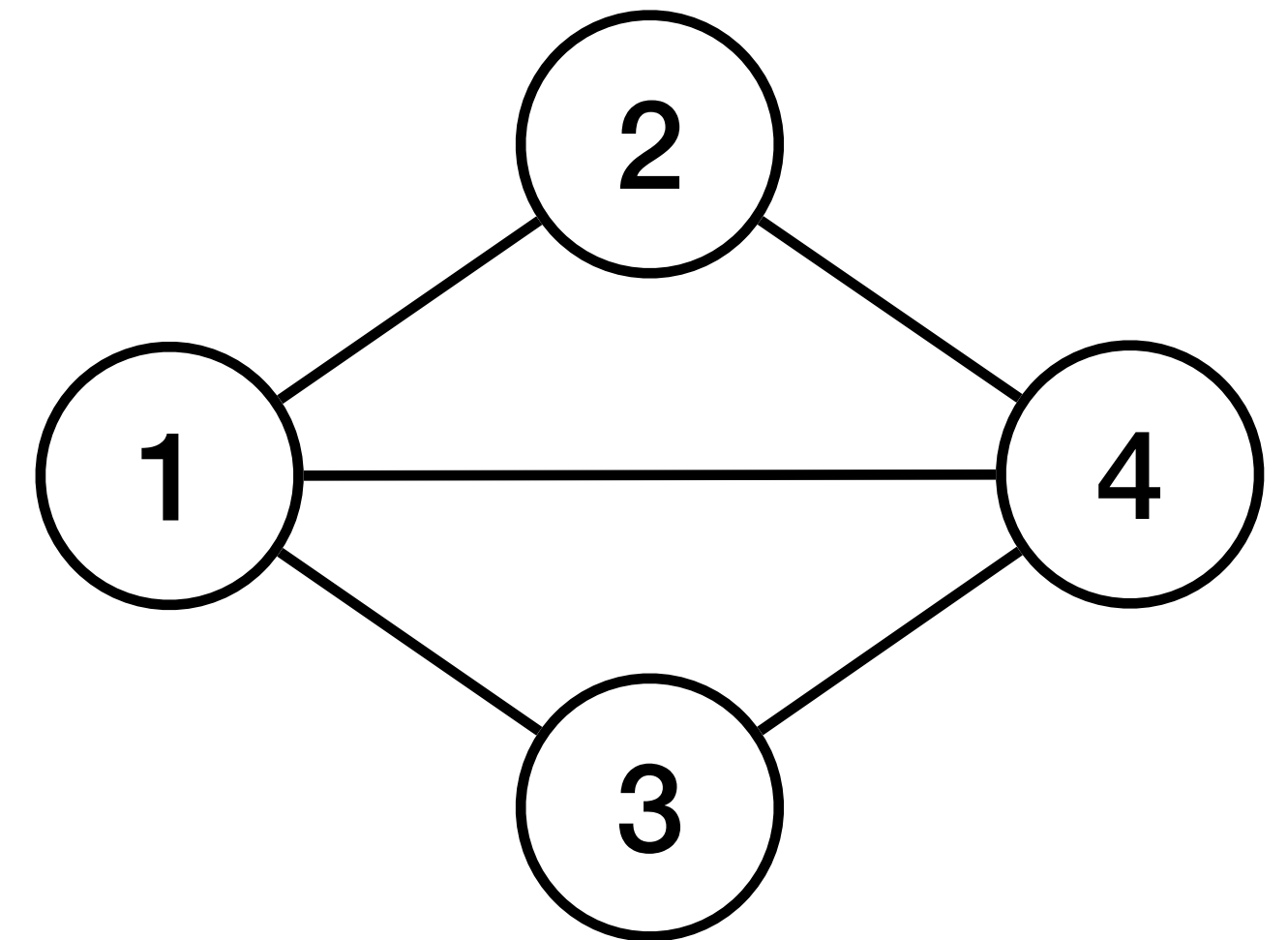
Graph colouring

- The decision problem “Can an undirected graph be coloured with k colours”? is also NP-complete
- Can the following graph be coloured with 3 colours?



Graph colouring

- Can the following graph be coloured with 3 colours?
- Allocate colours to vertices such that
 - Every vertex has one colour
 - At least one colour per vertex
 - At most one colour per vertex
 - Adjacent vertices have different colours



Variables

		Colour		
		Red	Green	Blue
Vertex	1	$x_{1,r}$	$x_{1,g}$	$x_{1,b}$
	2	$x_{2,r}$	$x_{2,g}$	$x_{2,b}$
	3	$x_{3,r}$	$x_{3,g}$	$x_{3,b}$
	4	$x_{4,r}$	$x_{4,g}$	$x_{4,b}$

Every vertex has one colour

At least...

$$(x_{1,r} \vee x_{1,g} \vee x_{1,b})$$

\wedge

$$(x_{2,r} \vee x_{2,g} \vee x_{2,b})$$

\wedge

$$(x_{3,r} \vee x_{3,g} \vee x_{3,b})$$

\wedge

$$(x_{4,r} \vee x_{4,g} \vee x_{4,b})$$

At most...

$$\neg(x_{1,r} \wedge x_{1,g}) \wedge \neg(x_{1,r} \wedge x_{1,b}) \wedge \neg(x_{1,g} \wedge x_{1,b})$$

\wedge

$$\neg(x_{2,r} \wedge x_{2,g}) \wedge \neg(x_{2,r} \wedge x_{2,b}) \wedge \neg(x_{2,g} \wedge x_{2,b})$$

\wedge

$$\neg(x_{3,r} \wedge x_{3,g}) \wedge \neg(x_{3,r} \wedge x_{3,b}) \wedge \neg(x_{3,g} \wedge x_{3,b})$$

\wedge

$$\neg(x_{4,r} \wedge x_{4,g}) \wedge \neg(x_{4,r} \wedge x_{4,b}) \wedge \neg(x_{4,g} \wedge x_{4,b})$$

Adjacent vertices have different colours

$$\neg(x_{1,r} \wedge x_{2,r}) \wedge \neg(x_{1,g} \wedge x_{2,g}) \wedge \neg(x_{1,b} \wedge x_{2,b})$$

$$\wedge$$

$$\neg(x_{1,r} \wedge x_{3,r}) \wedge \neg(x_{1,g} \wedge x_{3,g}) \wedge \neg(x_{1,b} \wedge x_{3,b})$$

$$\wedge$$

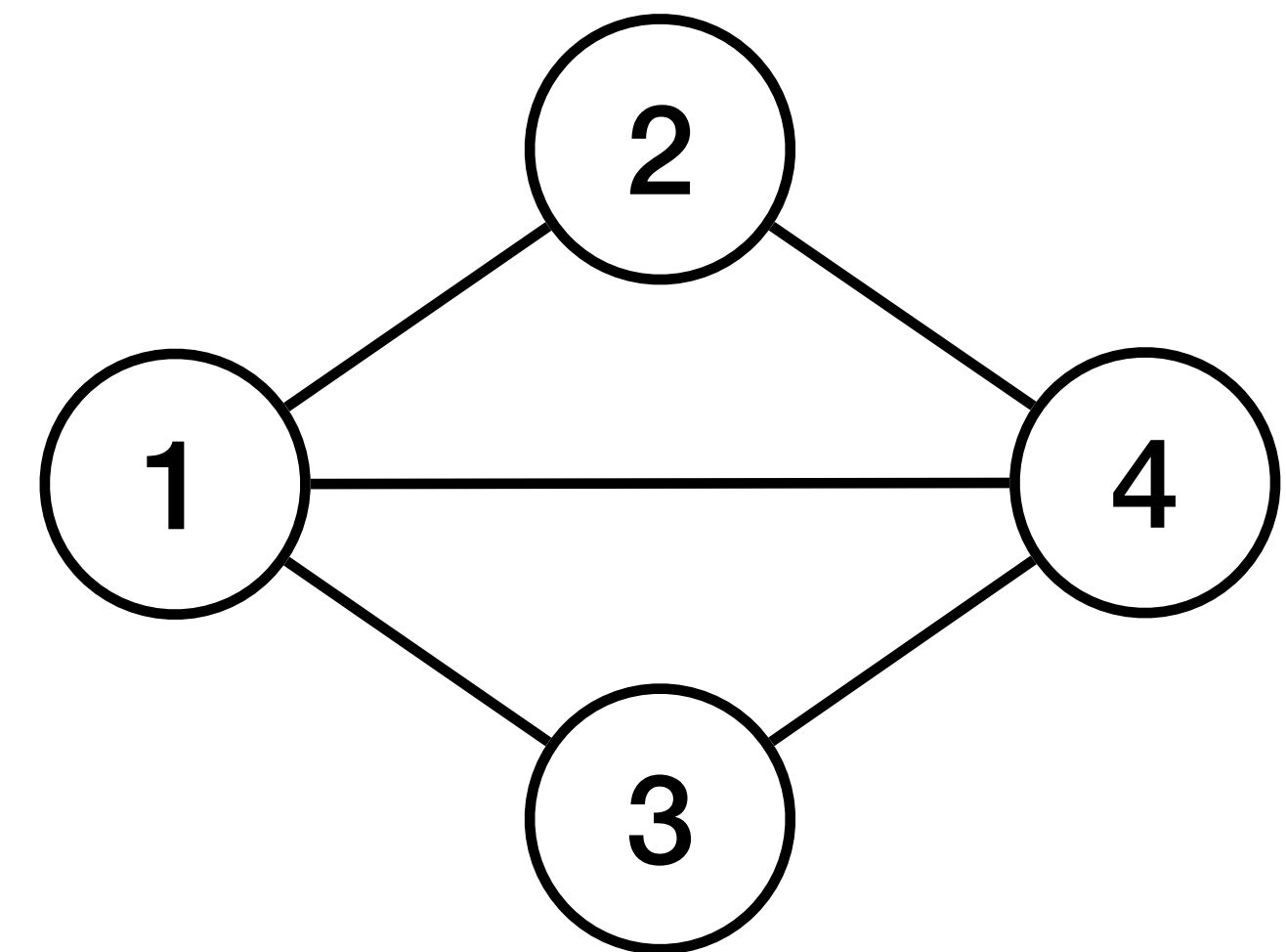
$$\neg(x_{1,r} \wedge x_{4,r}) \wedge \neg(x_{1,g} \wedge x_{4,g}) \wedge \neg(x_{1,b} \wedge x_{4,b})$$

$$\wedge$$

$$\neg(x_{2,r} \wedge x_{4,r}) \wedge \neg(x_{2,g} \wedge x_{4,g}) \wedge \neg(x_{2,b} \wedge x_{4,b})$$

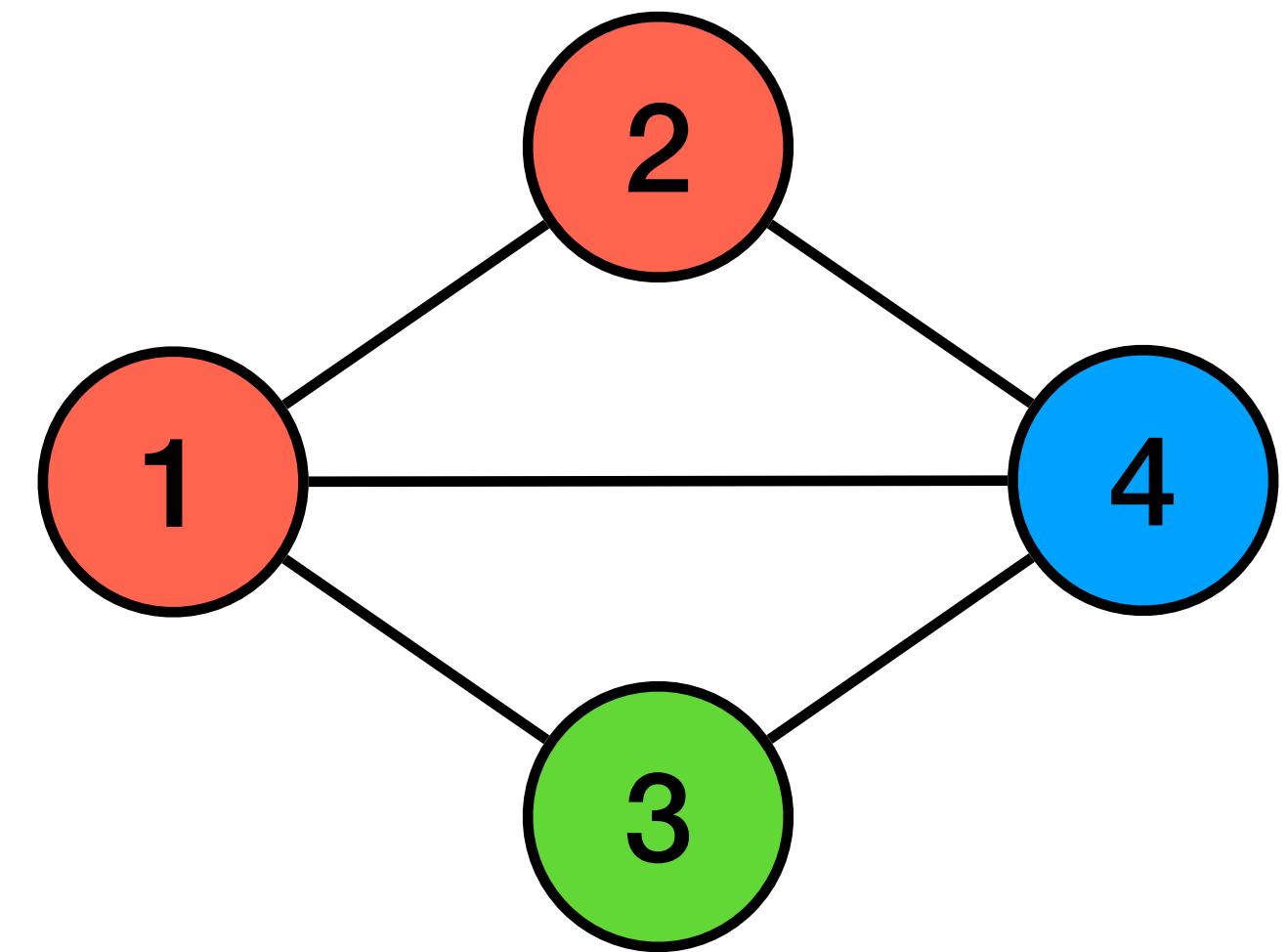
$$\wedge$$

$$\neg(x_{3,r} \wedge x_{4,r}) \wedge \neg(x_{3,g} \wedge x_{4,g}) \wedge \neg(x_{3,b} \wedge x_{4,b})$$



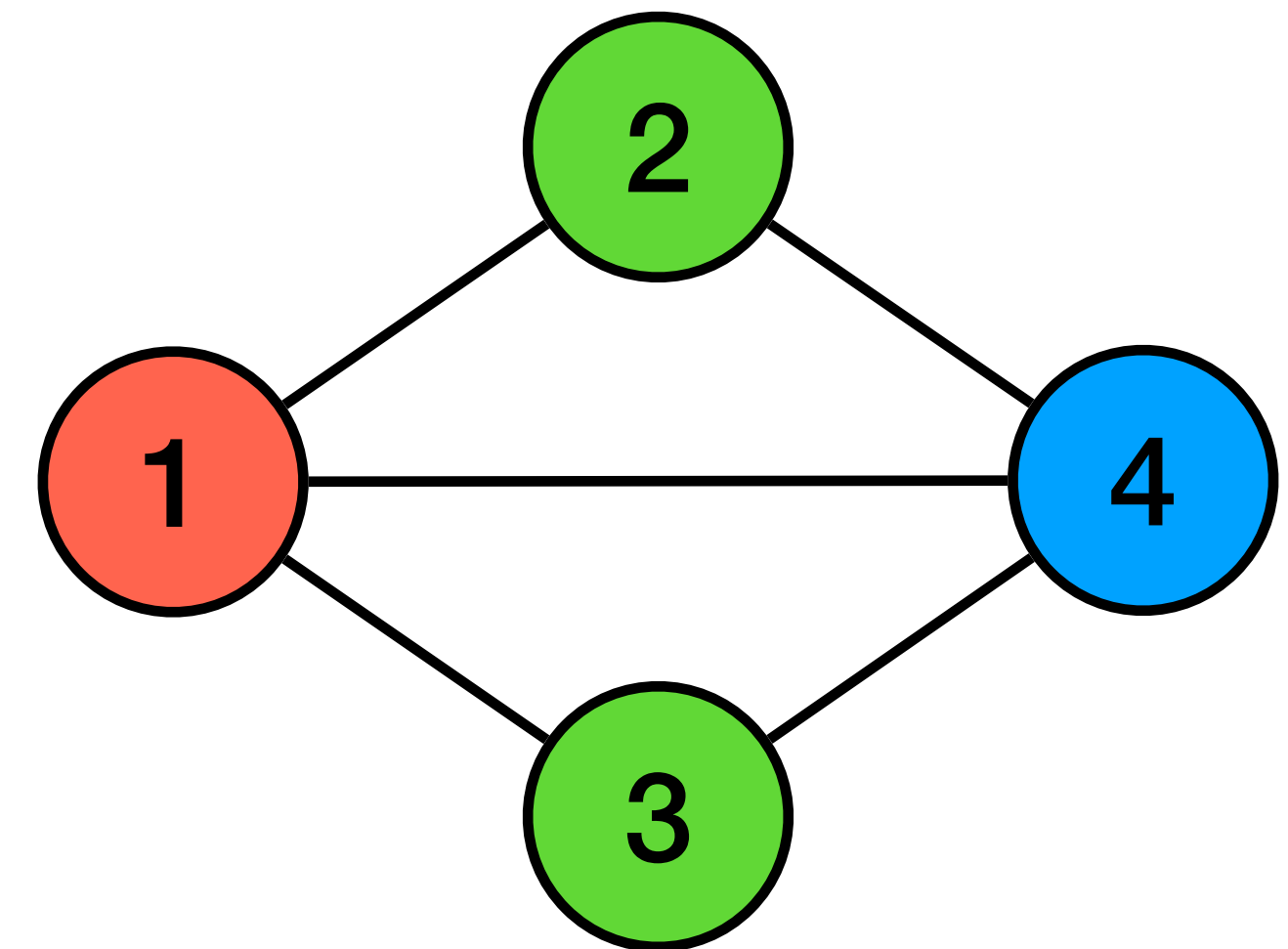
Brute force

1	2	3	4	ϕ
R	R	R	R	X
R	R	R	G	X
R	R	R	B	X
R	R	G	R	X
R	R	G	G	X
R	R	G	B	X



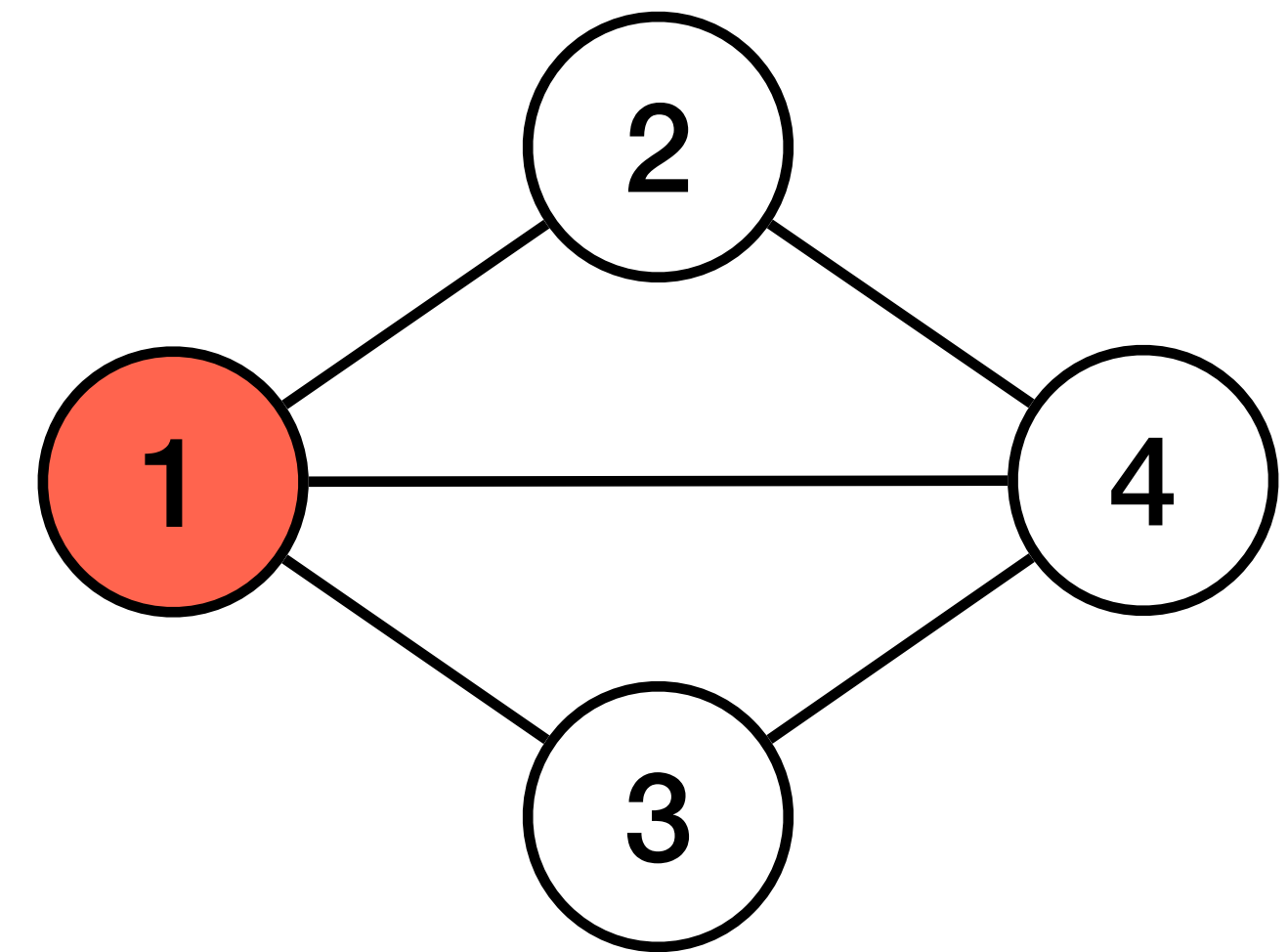
Brute force

1	2	3	4	ϕ
R	R	R	R	X
R	R	R	G	X
R	R	R	B	X
R	R	G	R	X
R	R	G	G	X
R	R	G	B	X
R	R	B	R	X
		...		
R	G	G	B	✓



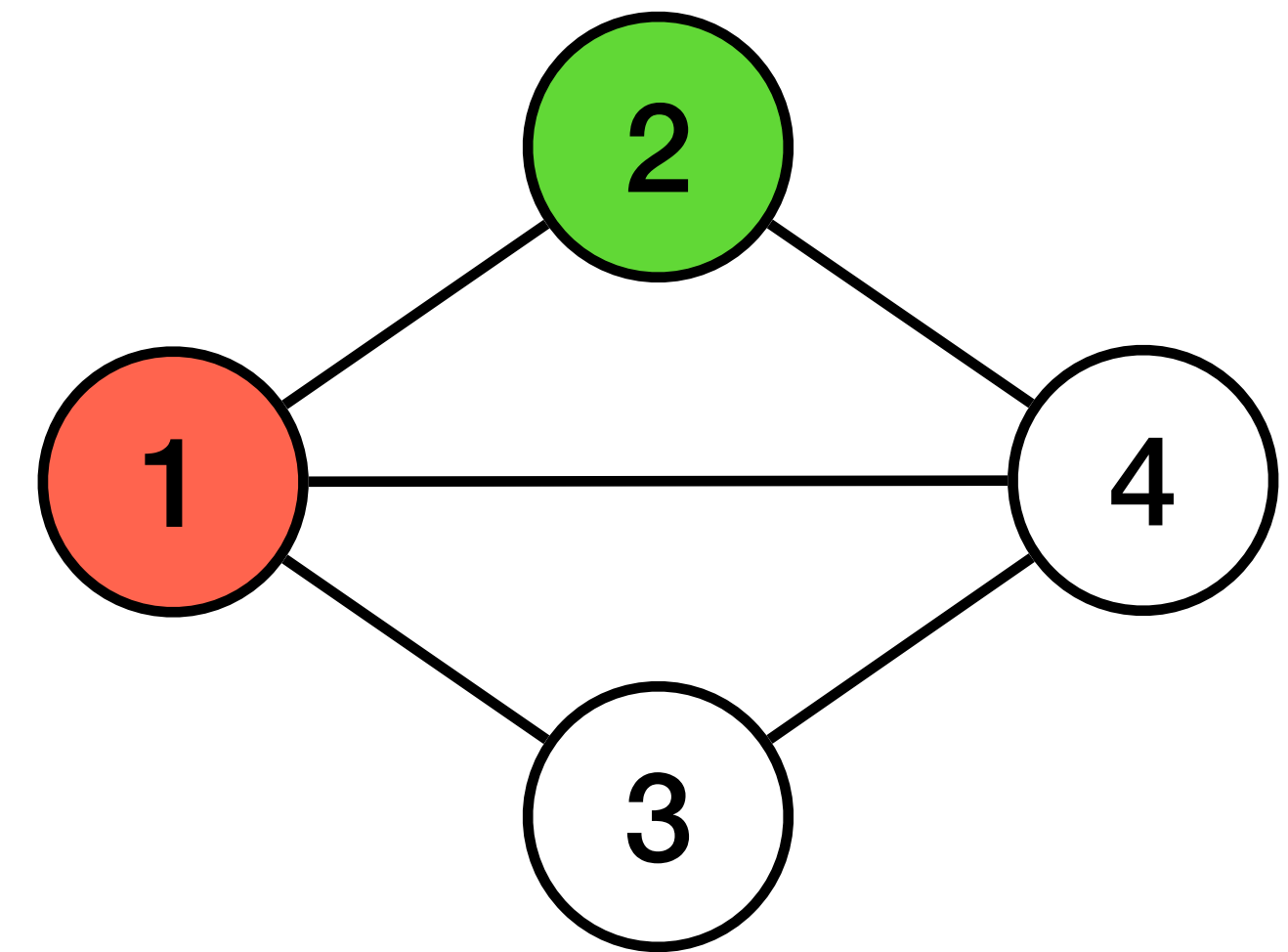
Backtracking

1	2	3	4	ϕ
R				✓



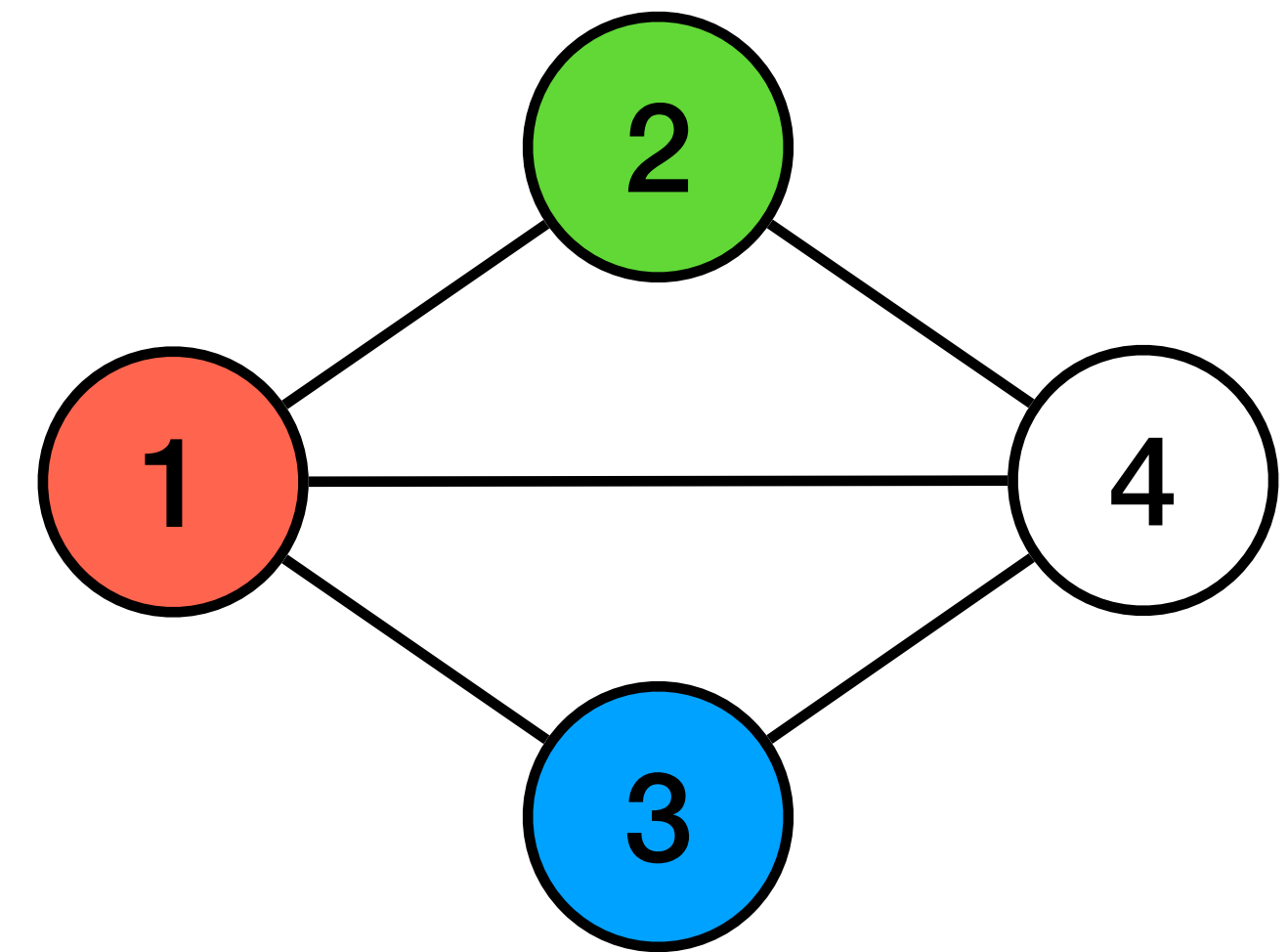
Backtracking

1	2	3	4	ϕ
R				✓
R	G			✓



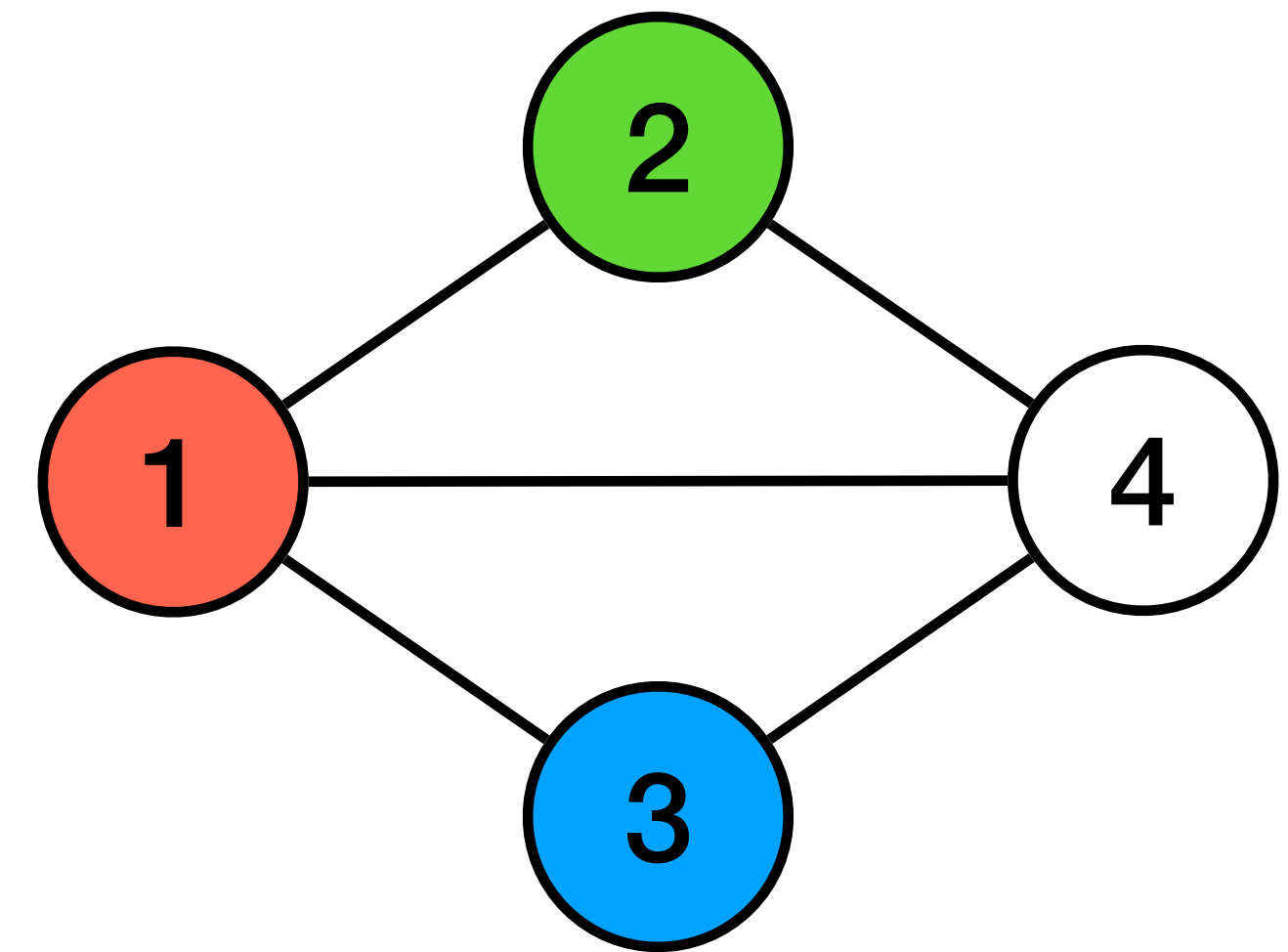
Backtracking

1	2	3	4	ϕ
R				✓
R	G			✓
R	G	B		✓



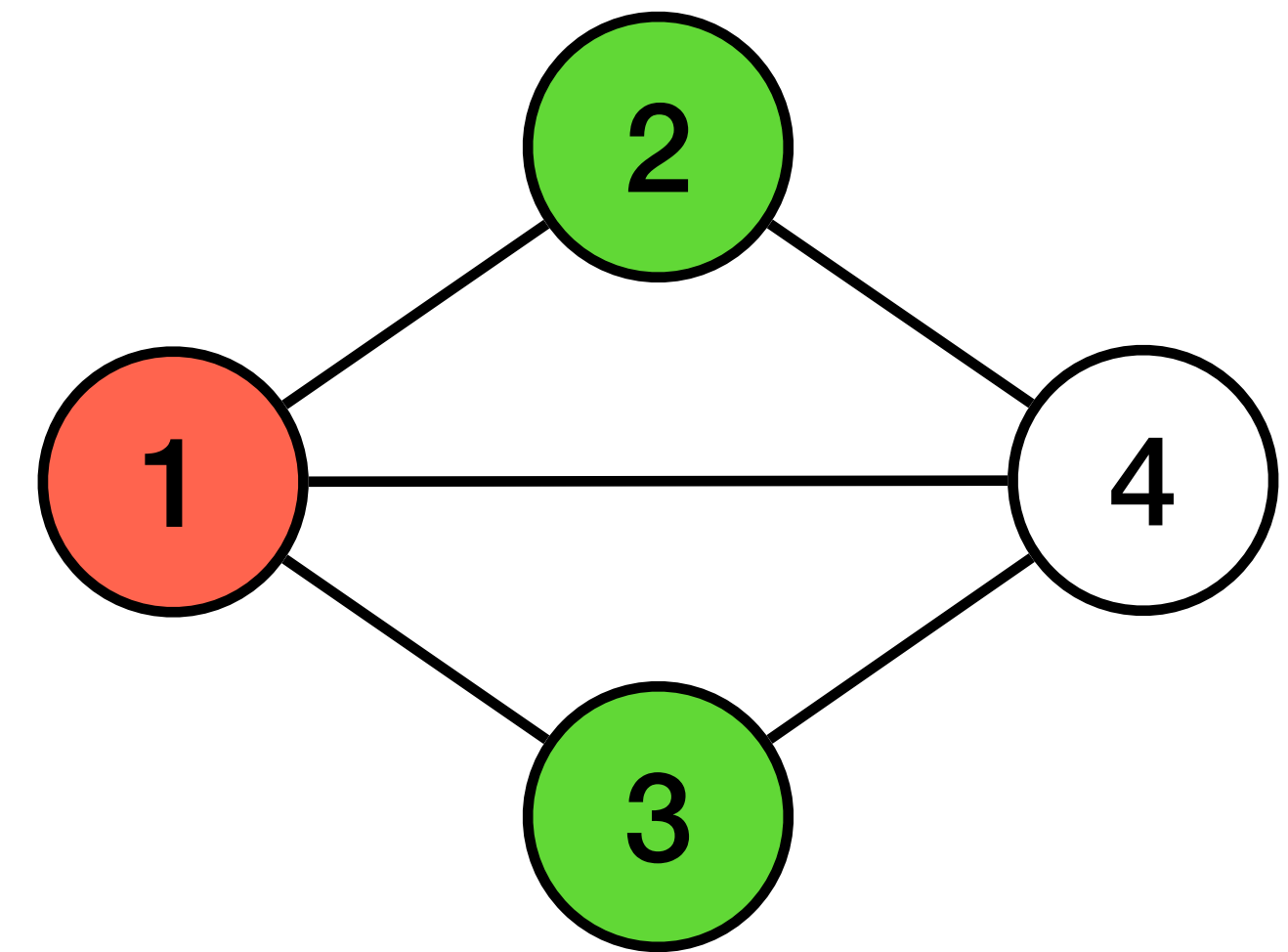
Backtracking

1	2	3	4	ϕ
R				✓
R	G			✓
R	G	B		✓
R	G	B	?	✗



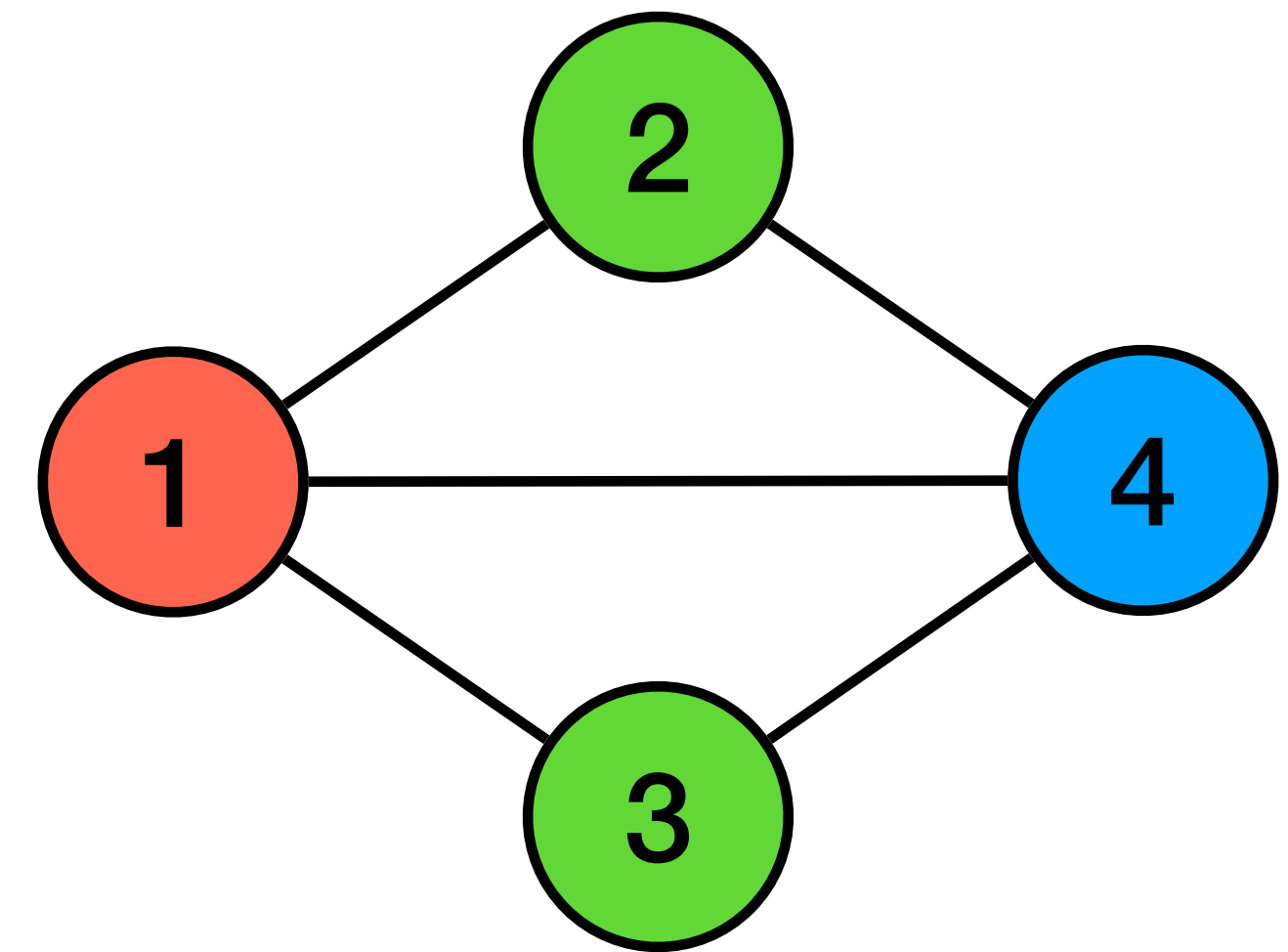
Backtracking

1	2	3	4	ϕ
R				✓
R	G			✓
R	G	B		✓
R	G	B	?	✗
R	G	G		✓



Backtracking

1	2	3	4	ϕ
R				✓
R	G			✓
R	G	B		✓
R	G	B	?	✗
R	G	G		✓
R	G	G	B	✓



DPLL algorithm

- Introduced by Davis, Putman, Logemann, and Loveland
- Backtracking algorithm that incrementally searches for a satisfiable assignment
- Works on propositional logic formulas in **conjunctive normal form** (CNF)

CNF

- A formula in CNF is a conjunction of clauses

$$(A \vee B) \wedge (B \vee \neg C \vee D) \wedge (\neg A \vee \neg B) \wedge (\neg A \vee \neg C \vee \neg D) \wedge A$$

- A **clause** is disjunction of literals
- A **literal** is a propositional variable or its negation
- Any formula can be converted to CNF by applying well-known rewrite rules
- Formulas in CNF can be represented in a compact way

$$\{ \{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\} \}$$

Conversion to CNF

$$\phi \leftrightarrow \psi \quad \Rightarrow \quad (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$$

$$\phi \rightarrow \psi \quad \Rightarrow \quad \neg\phi \vee \psi$$

$$\neg\neg\phi \quad \Rightarrow \quad \phi$$

$$\neg(\phi \vee \psi) \quad \Rightarrow \quad \neg\phi \wedge \neg\psi$$

$$\neg(\phi \wedge \psi) \quad \Rightarrow \quad \neg\phi \vee \neg\psi$$

$$\phi \vee (\psi \wedge \theta) \quad \Rightarrow \quad (\phi \vee \psi) \wedge (\phi \vee \theta)$$

$$(\psi \wedge \theta) \vee \phi \quad \Rightarrow \quad (\psi \vee \phi) \wedge (\theta \vee \phi)$$

DPLL algorithm

- Start from an empty assignment
- Repeat the following steps
 - **Deduce** the value of literals and perform **unit propagation**
 - If nothing to deduce, then **guess** the value of literal
 - If a clause becomes empty we reached a contradiction (aka **conflict**)
 - **Backtrack** to last “open” guess and try opposite value
 - If the formula becomes empty we reached a satisfiable assignment
- If the search terminates without finding a satisfiable assignment the formula is unsatisfiable

Unit propagation

- A **unit clause** contains only one literal

$$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$$

- We **deduce** that literal must be true for the formula to be satisfied
- And **unit propagation** can be performed
 - **Discard** all clauses containing the literal
 - **Discard** the literal complement from all clauses

Unit propagation

- A **unit clause** contains only one literal

$$\{\{\cancel{A, B}\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$$

- We **deduce** that literal must be true for the formula to be satisfied
- And **unit propagation** can be performed
 - **Discard** all clauses containing the literal
 - **Discard** the literal complement from all clauses

Unit propagation

- A **unit clause** contains only one literal

$$\{\{\cancel{A, B}\}, \{B, \bar{C}, D\}, \{\cancel{A}, \bar{B}\}, \{\cancel{A}, \bar{C}, \bar{D}\}, \{A\}\}$$

- We **deduce** that literal must be true for the formula to be satisfied
- And **unit propagation** can be performed
 - **Discard** all clauses containing the literal
 - **Discard** the literal complement from all clauses

$$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$$

Example SAT

A	B	C	D

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

Example SAT

A	B	C	D	
1				deduce A

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

Example SAT

A	B	C	D	
1				deduce A

~~$\{A, B\}$~~ , $\{B, \bar{C}, D\}$, ~~$\{\bar{A}, \bar{B}\}$~~ , ~~$\{\bar{A}, \bar{C}, \bar{D}\}$~~ , $\{A\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{\cancel{B}, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}
1	0	1		guess C

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}
1	0	1		guess C

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$

$\{\{D\}, \{\bar{D}\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}
1	0	1		guess C
1	0	1	1	deduce D

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$

$\{\{D\}, \{\bar{D}\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}
1	0	1		guess C
1	0	1	1	deduce D

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$

$\{\{D\}, \{\bar{D}\}\}$

$\{\{\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}
1	0	1		guess C
1	0	1	1	deduce D
1	0	0		guess \bar{C}

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$

$\{\{D\}, \{\bar{D}\}\}$

$\{\{\}\}$

Example SAT

A	B	C	D	
1				deduce A
1	0			deduce \bar{B}
1	0	1		guess C
1	0	1	1	deduce D
1	0	0		guess \bar{C}

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

$\{\{B, \bar{C}, D\}, \{\bar{B}\}, \{\bar{C}, \bar{D}\}\}$

~~$\{\{\bar{C}, D\}, \{\bar{C}, \bar{D}\}\}$~~

$\{\{D\}, \{\bar{D}\}\}$

$\{\{\}\}$

$\{\}$

Example UNSAT

A	B	C

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

Example UNSAT

A	B	C	
1			guess A

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

Example UNSAT

A	B	C	
1			guess A
1	1		deduce B

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \{C\}\}$

Example UNSAT

A	B	C	
1			guess A
1	1		deduce B
1	1	1	deduce C

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \textcircled{C}\}$

$\{\{\}\}$

Example UNSAT

A	B	C	
1			guess A
1	1		deduce B
1	1	1	deduce C
0			guess \bar{A}

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \{C\}\}$

$\{\{\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

Example UNSAT

A	B	C	
1			guess A
1	1		deduce B
1	1	1	deduce C
0			guess \bar{A}
0	1		deduce B

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \{C\}\}$

$\{\{\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \{C\}\}$

Example UNSAT

A	B	C	
1			guess A
1	1		deduce B
1	1	1	deduce C
0			guess \bar{A}
0	1		deduce B
0	1	1	deduce C

$\{\{A, B\}, \{\bar{A}, B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \{C\}\}$

$\{\{\}\}$

$\{\{B\}, \{\bar{B}, \bar{C}\}, \{\bar{B}, C\}\}$

$\{\{\bar{C}\}, \{C\}\}$

$\{\{\}\}$

CDCL algorithm

- Conflict-Driven Clause Learning improves DPLL significantly
- CDCL keeps an implication graph with the decisions that led to a conflict
- From this graph it can “learn” a new clause that rules out that conflict
- Backtrack directly jumps (non-chronologically) to a decision where the conflict can still be avoided
- CDCL is at the basis of state-of-the-art SAT solvers

DIMACS

- Standard textual format for CNF
- Used by most SAT solvers
- Starts by declaring the number of variables and the number of clauses
- Each clause is described by a sequence of literals ended by 0
- Each literal is either a positive number identifying a variable or its negation

DIMACS

$\{\{A, B\}, \{B, \bar{C}, D\}, \{\bar{A}, \bar{B}\}, \{\bar{A}, \bar{C}, \bar{D}\}, \{A\}\}$

```
c example
p cnf 4 5
1 2 0
2 -3 4 0
-1 -2 0
-1 -3 -4 0
1 0
```

satcompetition.github.io



SAT Competition 2025

SAT Competition 2025 is a competitive event for solvers of the Boolean Satisfiability (SAT) problem. The competition is organized as a satellite event to the [SAT Conference 2025](#) and continues the series of the annual [SAT Competitions and SAT-Races / Challenges](#).

Objective

The area of SAT Solving has seen tremendous progress over the last years. Many problems that seemed to be out of reach a decade ago can now be handled routinely. Besides new algorithms and better heuristics, refined implementation techniques turned out to be vital for this success. To keep up the driving force in improving SAT solvers, we want to motivate implementers to present their work to a broader audience and to compare it with that of others. Researchers from both academia and industry are invited to submit their solvers and benchmarks.

News

- **2025-08-25** The Benchmarks and Solvers are available in the [Downloads Section](#).
- **2025-08-25** The Proceedings of the SAT Competition 2025 is available online [here](#).
- **2025-08-16** The slides of the SAT Competition 2025 results are available [here](#).
- **2025-02-16** The SAT Competition 2025 will be hosted on the BenchCloud instance of the SoSy Lab in Munich for the first time. This results in changes to the [submission and resource limits](#).

Overview

Competition Tracks

Solver Submission

- [Output Format](#)
- [BenchCloud Cluster](#)
- [AWS Cloud](#)

Benchmark Submission

Downloads

Organizers

Beyond SAT solving

- SAT solving deals only with propositional logic
- **Satisfiability Modulo Theories** (SMT) generalizes SAT
- Supports first-order logic with symbol interpretations restricted by theories
- Undecidable in general, but decidable under some theories and fragments
 - Linear integer arithmetic
 - Reals
 - Arrays
 - ...



Let's do it with Z3!

