# SAT – Propositional Satisfiability

SAT:
*Input:* a propositional formula $\varphi$ in CNF
*Question:* is $\varphi$ satisfiable?

Example (satisfiable):

$$(x_1 \vee x_2) \wedge (x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_1 \vee x_4) \wedge (x_4)$$

Example (unsatisfiable):

$$(x_1 \vee x_2) \wedge (\neg x_2) \wedge (\neg x_1) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_4)$$

# SAT solvers & encodings

Best algorithms take $2^n$ time in the worst-case

but are very efficient in many cases (instances with millions of variables solved in seconds)

> *Progress on the engineering [of SAT solvers] has been nothing short of spectacular.* – Moshe Vardi

# SAT solvers & encodings

Best algorithms take $2^n$ time in the worst-case

but are very efficient in many cases (instances with millions of variables solved in seconds)

> *Progress on the engineering [of SAT solvers] has been nothing short of spectacular.* – Moshe Vardi

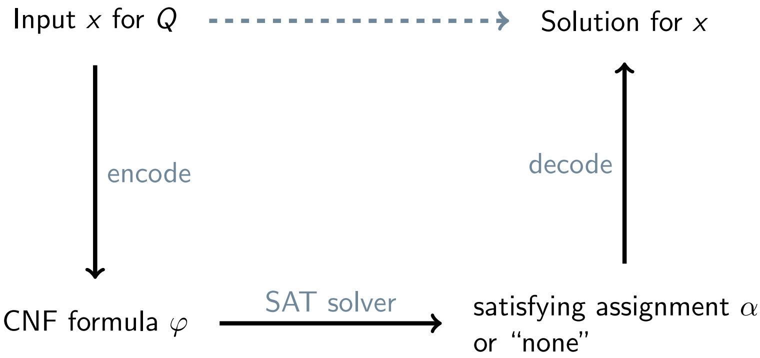
**Strategy to solve your favorite NP-complete problem $Q$:**

1. encode instance of $Q$ into instance $\varphi$ of SAT (in poly-time)
2. call a SAT solver on $\varphi$
3. decode solution into a solution for $Q$

Example problem: does a graph $G$ have a clique of size $\geq m$?

# SAT encoding (in a picture)

# SAT encoding (in a picture)



# Using SAT encodings

Given any graph $G$ and a natural number $k$, we build a formula $F(G, k)$ such that:

# Using SAT encodings

Given any graph $G$ and a natural number $k$, we build a formula $F(G, k)$ such that:

**Idea**

The formula $F(G, k)$ is true if and only if $G$ has a parameter with value $k$.

Here $k$ can be any of the following graph parameters

- Treewidth (Samer & Veith 2009),
- Cliquewidth (Heule & Szeider 2013),
- Branchwidth (current work : Apply this approach for larger graphs using SAT for local improvement), ...

# Counting

- How to count something using *SAT* formula?
- Various techniques (Bjork 2009)
  - Unary encoding
  - Binary encoding
  - onehot encoding

# Cardinality Constraints using Unary counting

- Let $L(v)$ be *True* when vertex $v$ is being counted.
- Let $C(i)$ be true when counter has value $i$
- We can count 2 using following formula

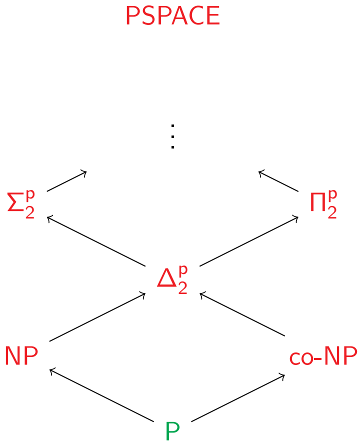
$$(L(1) \implies C(1))$$

# Cardinality Constraints using Unary counting

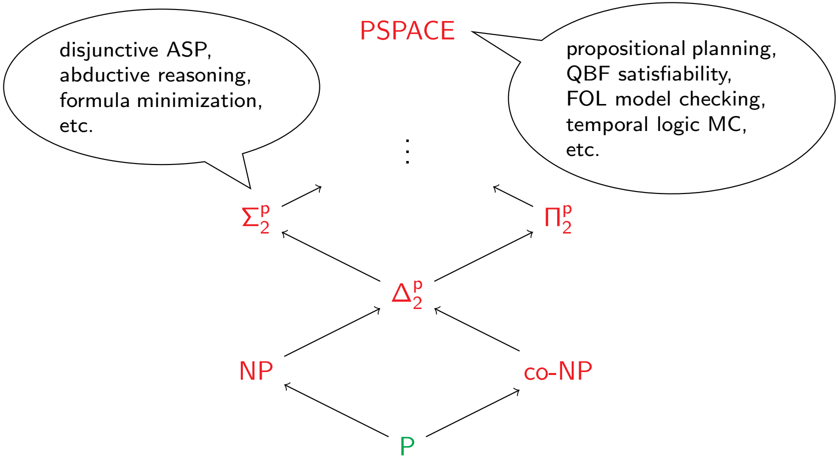
- Let $L(v)$ be *True* when vertex $v$ is being counted.
- Let $C(i)$ be true when counter has value $i$
- We can count 2 using following formula

$$(L(1) \implies C(1)) \wedge (C(1) \wedge L(2) \implies C(2))$$

# Polynomial Hierarchy: Problems ‘Beyond NP’



# Polynomial Hierarchy: Problems 'Beyond NP'



# Other kinds of reductions

Example problem: how big is the largest clique in a graph $G$?

Turing reductions: call a SAT solver multiple times

Example problem: find a subset of clauses that is *minimally unsatisfiable* (removing any clause makes it satisfiable) – MUS

# Other kinds of reductions

Example problem: how big is the largest clique in a graph $G$?

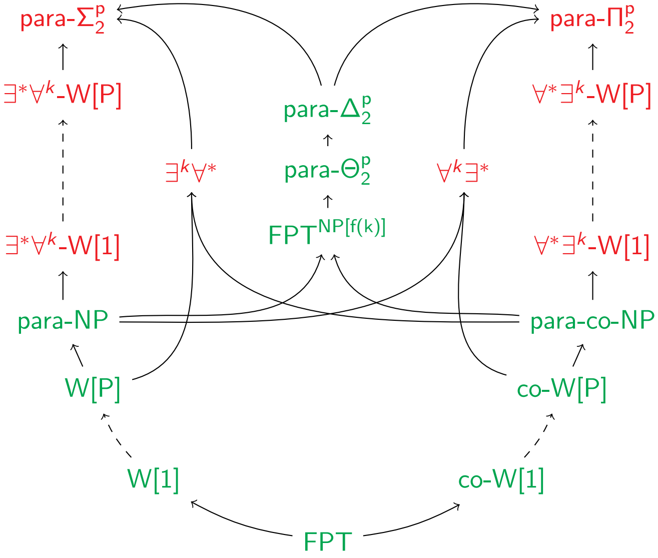
Turing reductions: call a SAT solver multiple times

Example problem: find a subset of clauses that is *minimally unsatisfiable* (removing any clause makes it satisfiable) – MUS

Fpt-reductions to SAT: allow fpt-time instead of poly-time

- Interesting for problems 'beyond NP'
- Example problem: find smallest logically equivalent CNF

# A new landscape



# References


M. Bjork.
Successful SAT encoding techniques.
*Journal on Satisfiability, Boolean Modeling and Computation, Addendum, IOS Press*, 2009.

M. J. H. Heule and S. Szeider.
A SAT approach to clique-width.
*ACM Trans. Comput. Logic*, 16(3):24:1–24:27, June 2015.

M. Samer and H. Veith.
Encoding treewidth into SAT.
In *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, pages 45–50, 2009.