# SAT

# SATisfiability

- A well-known decision problem: given a propositional formula, is it SATisfiable?
- Propositional Formula
  - A formula in propositional logic composed of Boolean (decision) variables and the logical operators.

    $\neg$ (negation, "not"), $\vee$ (disjunction, "or"), $\wedge$ (conjunction, "and")

    $\rightarrow$ (implication, $p \rightarrow q \equiv \neg p \vee q$))

    $\leftrightarrow$ (bi-implication, $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$)
  - Examples
    - $\neg p$
    - $(p \vee q)$
    - $\neg(p \wedge q)$
    - $p \wedge \neg(p \vee q)$
    - $(p \wedge q) \vee (r \vee \neg q)$

# Satisfiability of a Propositional Formula

- A propositional formula is satisfiable iff it is possible to find a truth assignment to the variables that yields the formula true.
  - $(p \vee q) \wedge (\neg p \vee \neg q) \wedge (p \vee \neg q)$ is SAT
    - $p = T$ and $q = F$
  - $(p \vee q) \wedge (\neg p \vee \neg q) \wedge (\neg p \vee q) \wedge (p \vee \neg q)$ is UNSAT

# Why SAT?

- The first proven NP-complete decision problem.
  - The Cook-Levin Theorem (1971).
  - The starting point of NP-completeness research.
- A theoretical standard for hard problems.
  - All problems in the complexity class NP, including a wide range of decision and optimization problems, are reducible to SAT in polynomial time (i.e. at most as difficult to solve as SAT).
  - SAT can be used to prove that a problem is NP-complete.
    - Show that the problem is in NP.
    - Show that SAT can be reduced to the problem in polynomial time.
- Widely used modeling framework for solving combinatorial decision problems.

# SAT is Everywhere!

- Artificial Intelligence
  - Automated theorem proving
  - Non-monotonic reasoning
  - Planning
  - Machine learning (verification and explanation of ML models)
  - …
- Successful applications in software engineering
  - Equivalence check in HW optimization (i.e. functional equivalence of the original and optimized combinational circuits)
  - Equivalence check in SW optimization (i.e. functional equivalence of original and optimized code)

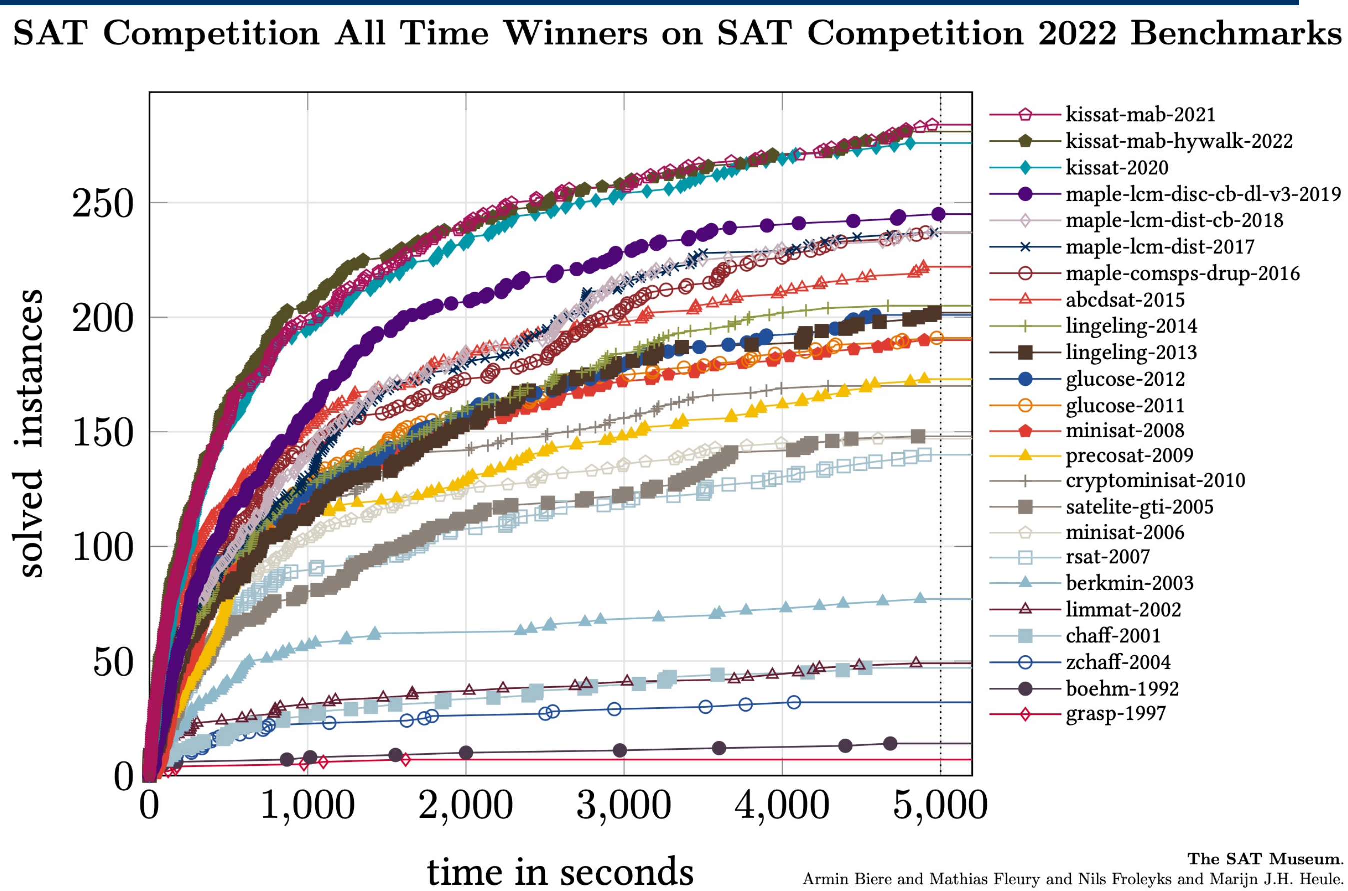
# Brute Force Approach

- Given a formula with $n$ variables, compute the values of the formula for all $2^n$ ways to choose values $T$ and $F$ for the variables.
  - The formula is SAT iff at least one of the $2^n$ cases yields true.
  - Simple, always works ☺
  - Exponential size in the number of the variables, impractical for big values of $n$ ☹
- We need dedicated SAT solvers for checking the satisfiability of a formula and finding a satisfying assignment.

# SAT Solvers

- Bad news ☹
  - Being NP-complete, it is generally believed that no algorithm that can efficiently solves SAT exists.
  - Resolving the question of whether SAT has a polynomial-time algorithm is equivalent to the P vs NP problem, which is a famous open problem in the theory of computing.
- Good news ☺
  - Current SAT solvers are successful for big formulas.
    - In mid 90's, SAT solvers could solve formulas with thousands of variables.
    - Nowadays, they can solve formulas with millions of variables and clauses!

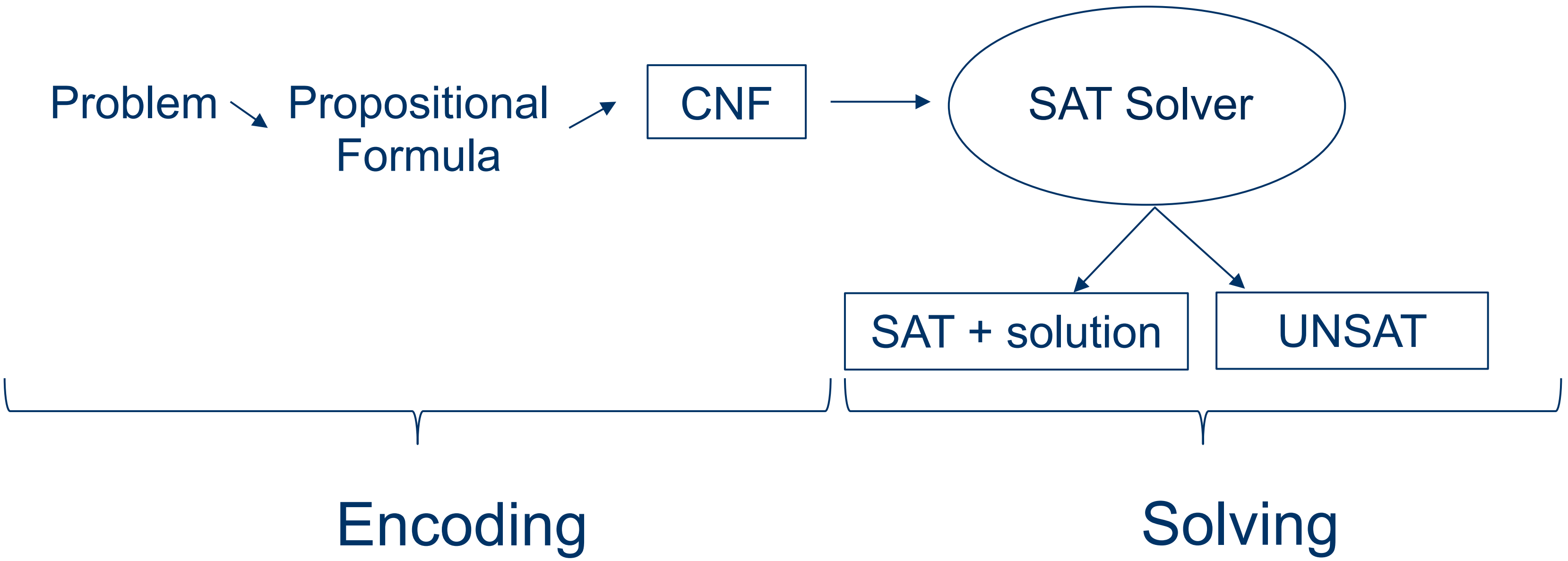
# SAT Solvers



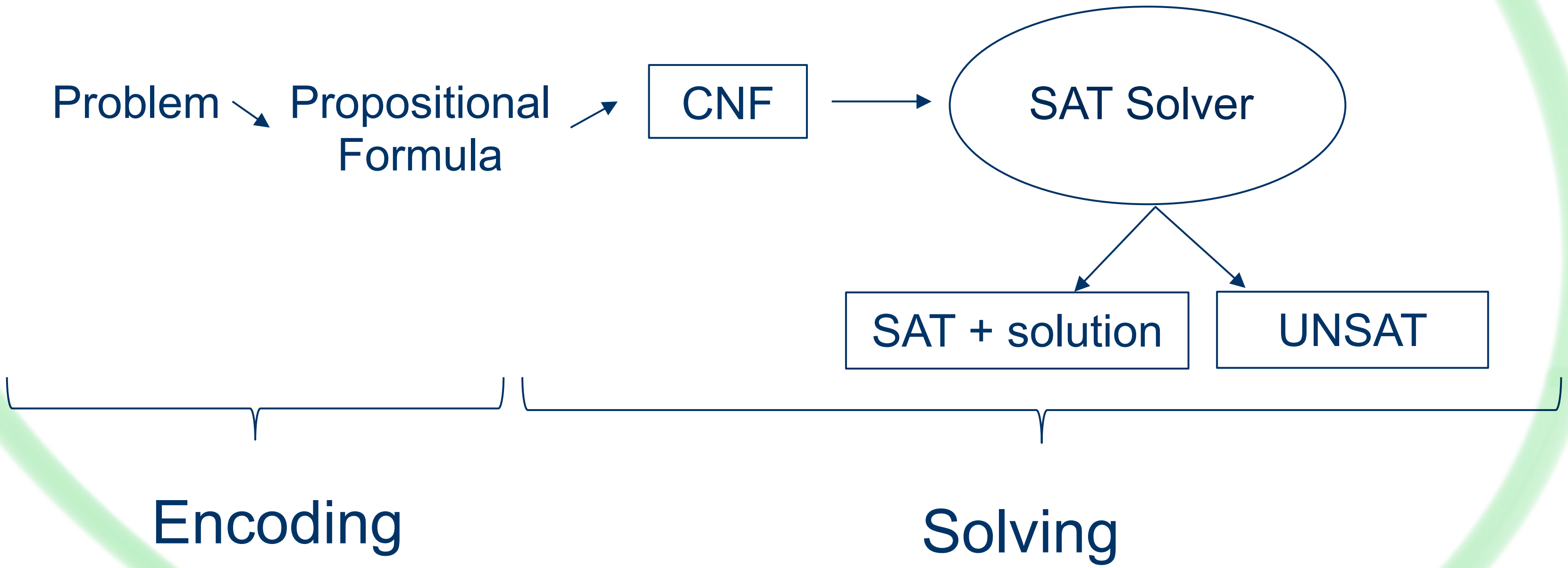

https://cca.informatik.uni-freiburg.de/satmuseum

**The SAT Museum.**
Armin Biere and Mathias Fleury and Nils Froleyks and Marijn J.H. Heule.
In *Proceedings 14th International. Workshop on Pragmatics of SAT (POS'23)*,
vol. 3545, CEUR Workshop Proceedings, pages 72-87, CEUR-WS.org 2023.
[ paper - bibtex - data - zenodo - ceur - workshop - proceedings ]

# Problem Solving with SAT



# Problem Solving with SAT



# Formula Satisfiability vs Validation

- A formula $f$ is **satisfiable** if there is *an assignment* of values to the variables under which $f$ evaluates to $T$.
  - Satisfiability is about finding a solution to a set of constraints.

- A formula $f$ is **valid** if $f$ always evaluates to $T$ for *any assignment* of values to the variables.
  - Validity is about finding a proof of a statement.
  - A formula $f$ is **valid** iff $\neg f$ is UNSAT.

# Formula Validation Using SAT

- Let's prove deMorgan's law:
  - $a \wedge b \equiv \overline{\overline{(a \wedge b)}} \equiv \overline{(\neg a \vee \neg b)}$
- We need to show that:
  - $a \wedge b \leftrightarrow \overline{(\neg a \vee \neg b)}$
- How?
  - $f_1 \leftrightarrow f_2$ iff $\neg(f_1 \leftrightarrow f_2)$ is UNSAT
- Show that $\overline{\left(a \wedge b \leftrightarrow \overline{(\neg a \vee \neg b)}\right)}$ is UNSAT.

$$\overline{\left(a \wedge b \longleftrightarrow \overline{(\neg a \vee \neg b)}\right)} \overset{?}{\equiv} \bot$$

# Encoding for Validation

- Let’s encode a logic puzzle. Consider the following clues:
  1. Good-natured tenured professors are dynamic.
  2. Grumpy student advisors play slot machines.
  3. Smokers wearing a cap are phlegmatic.
  4. Comical student advisors are professors.
  5. Smoking untenured members are nervous.
  6. Phlegmatic tenured members wearing caps are comical.
  7. Student advisors who are not stock market players are scholars.
  8. Relaxed student advisors are creative.
  9. Creative scholars who do not play slot machines wear caps.
  10. Nervous smokers plat slot machines.
  11. Student advisors who play slot machines do not smoke.
  12. Creative good-natured stock market players wear caps.
- Can we conclude that no student advisor is smoking?

# Propositional Formula

- Give a name for every notion to be formalized.

| Name | Meaning | Opposite |
|---|---|---|
| $a$ | good-natured | grumpy |
| $b$ | tenured | |
| $c$ | professor | |
| $d$ | dynamic | phlegmatic |
| $e$ | wearing a cap | |
| $f$ | smoke | |
| $g$ | comical | |
| $h$ | relaxed | nervous |
| $i$ | play stock market | |
| $j$ | scholar | |
| $k$ | creative | |
| $l$ | play slot machine | |
| $m$ | student advisor | |

# Propositional Formula

- Write down each property as a formula. E.g.,

$$\underbrace{\text{Good-natured}}_{a}\ \underbrace{\text{tenured}}_{b}\ \underbrace{\text{professors}}_{c}\ \text{are}\ \underbrace{\text{dynamic}}_{d}$$

leads to $(a \wedge b \wedge c) \rightarrow d$.

# Propositional Formula

1. $(a \wedge b \wedge c) \rightarrow d$
2. $(\neg a \wedge m) \rightarrow l$
3. $(f \wedge e) \rightarrow \neg d$
4. $(g \wedge m) \rightarrow c$
5. $(f \wedge \neg b) \rightarrow \neg h$
6. $(\neg d \wedge b \wedge e) \rightarrow g$
7. $(\neg i \wedge m) \rightarrow j$
8. $(h \wedge m) \rightarrow k$
9. $(k \wedge j \wedge \neg l) \rightarrow e$
10. $(\neg h \wedge f) \rightarrow l$
11. $(l \wedge m) \rightarrow \neg f$
12. $(k \wedge a \wedge i) \rightarrow e$

$\neg(m \wedge f)$?

# Propositional Formula

- Prove that the following formula is UNSAT.

$$
\begin{array}{ll}
(a \wedge b \wedge c) \rightarrow d & \wedge \\
(\neg a \wedge m) \rightarrow l & \wedge \\
(f \wedge e) \rightarrow \neg d & \wedge \\
(g \wedge m) \rightarrow c & \wedge \\
(f \wedge \neg b) \rightarrow \neg h & \wedge \\
(\neg d \wedge b \wedge e) \rightarrow g & \wedge \quad m \wedge f \\
(\neg i \wedge m) \rightarrow j & \wedge \\
(h \wedge m) \rightarrow k & \wedge \\
(k \wedge j \wedge \neg l) \rightarrow e & \wedge \\
(\neg h \wedge f) \rightarrow l & \wedge \\
(l \wedge m) \rightarrow \neg f & \wedge \\
(k \wedge a \wedge i) \rightarrow e & \wedge
\end{array}
$$

# Encoding for Satisfaction

- Can we assign a color to the regions such that adjacent regions have different colors?

# Graph Coloring in SAT

- Can we assign a color to the regions such that adjacent regions have different colors?

- Variables
  - A Boolean variable $x_{ic}$ for each possible color $c \in \{1, \ldots, k\}$ and region $v_i \in V$.
    - $x_{ic} = T$ means region $i$ is assigned to the color $c$.

# Graph Coloring in SAT

- Constraints
  - Each region gets at least one color.
    - $x_{i1} \vee x_{i2} \vee \cdots \vee x_{ik}$ for all $v_i \in V$

      for all $v_i \in V$

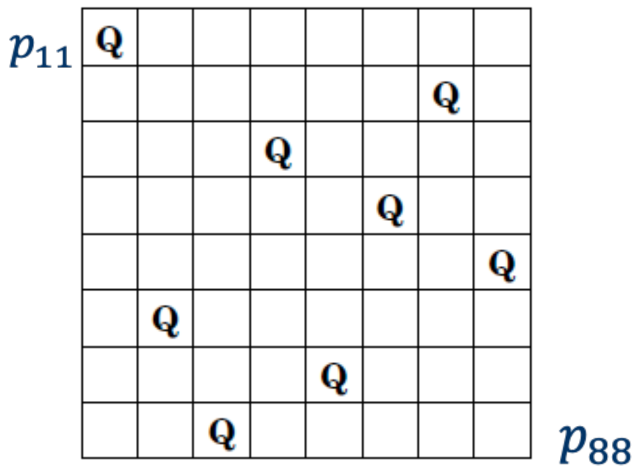
      $$\bigvee_{1 \le c \le k} x_{ic}$$
  - Neighboring regions take different colors.
    - $\neg(x_{ic} \wedge x_{jc})$ for all $(v_i, v_j) \in E$ and for all $c \in \{1, \ldots, k\}$

      for all $(v_i, v_j) \in E$

      $$\bigwedge_{1 \le c \le k} \neg(x_{ic} \wedge x_{jc})$$
  - Every region gets at most one color ?

# N-Queens in SAT

- Variables




  - A Boolean variable $p_{ij}$ for each $nxn$ position on the board.
    - $p_{ij} = T$ means there is a queen on row $i$ and column $j$.

# N-Queens in SAT

- Constraints

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_{11}$ | $p_{12}$ | $p_{13}$ | $p_{14}$ | $p_{15}$ | $p_{16}$ | $p_{17}$ | $p_{18}$ |
| $p_{21}$ | $p_{22}$ | $p_{23}$ | $p_{24}$ | $p_{25}$ | $p_{26}$ | $p_{27}$ | $p_{28}$ |
| $p_{31}$ | $p_{32}$ | $p_{33}$ | $p_{34}$ | $p_{35}$ | $p_{36}$ | $p_{37}$ | $p_{38}$ |
| $p_{41}$ | $p_{42}$ | $p_{43}$ | $p_{44}$ | $p_{45}$ | $p_{46}$ | $p_{47}$ | $p_{48}$ |
| $p_{51}$ | $p_{52}$ | $p_{53}$ | $p_{54}$ | $p_{55}$ | $p_{56}$ | $p_{57}$ | $p_{58}$ |
| $p_{61}$ | $p_{62}$ | $p_{63}$ | $p_{64}$ | $p_{65}$ | $p_{66}$ | $p_{67}$ | $p_{68}$ |
| $p_{71}$ | $p_{72}$ | $p_{73}$ | $p_{74}$ | $p_{75}$ | $p_{76}$ | $p_{77}$ | $p_{78}$ |
| $p_{81}$ | $p_{82}$ | $p_{83}$ | $p_{84}$ | $p_{85}$ | $p_{86}$ | $p_{87}$ | $p_{88}$ |

  - Exactly one queen on each row.

# N-Queens in SAT

- At least one queen on row $i$
  - $p_{i1} \vee p_{i2} \vee \cdots \vee p_{in}$ for all $i \in N$

    for all $i \in N$

$$\bigvee_{1 \le j \le n} p_{ij}$$

- At most one queen on row $i$
  - $\neg(p_{ij} \wedge p_{ik})$ for all $0 < j < k \le n$ for all $i \in N$

    for all $i \in N$

$$\bigwedge_{0 < j < k \le n} \neg(p_{ij} \wedge p_{ik})$$

# N-Queens in SAT

- Constraints

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_{11}$ | $p_{12}$ | $p_{13}$ | $p_{14}$ | $p_{15}$ | $p_{16}$ | $p_{17}$ | $p_{18}$ |
| $p_{21}$ | $p_{22}$ | $p_{23}$ | $p_{24}$ | $p_{25}$ | $p_{26}$ | $p_{27}$ | $p_{28}$ |
| $p_{31}$ | $p_{32}$ | $p_{33}$ | $p_{34}$ | $p_{35}$ | $p_{36}$ | $p_{37}$ | $p_{38}$ |
| $p_{41}$ | $p_{42}$ | $p_{43}$ | $p_{44}$ | $p_{45}$ | $p_{46}$ | $p_{47}$ | $p_{48}$ |
| $p_{51}$ | $p_{52}$ | $p_{53}$ | $p_{54}$ | $p_{55}$ | $p_{56}$ | $p_{57}$ | $p_{58}$ |
| $p_{61}$ | $p_{62}$ | $p_{63}$ | $p_{64}$ | $p_{65}$ | $p_{66}$ | $p_{67}$ | $p_{68}$ |
| $p_{71}$ | $p_{72}$ | $p_{73}$ | $p_{74}$ | $p_{75}$ | $p_{76}$ | $p_{77}$ | $p_{78}$ |
| $p_{81}$ | $p_{82}$ | $p_{83}$ | $p_{84}$ | $p_{85}$ | $p_{86}$ | $p_{87}$ | $p_{88}$ |

  - Exactly one queen on each column.

# N-Queens in SAT

- Column constraints are the same as the row constraints with $i$ and $j$ swapped.
  - At least one queen on each row and column

$$\bigwedge_{1 \le i \le n} \bigvee_{1 \le j \le n} p_{ij} \quad \bigwedge_{1 \le j \le n} \bigvee_{1 \le i \le n} p_{ij}$$

  - At most one queen on each row and column

$$\bigwedge_{1 \le i \le n} \bigwedge_{0 < j < k \le n} \neg(p_{ij} \wedge p_{ik}) \quad \bigwedge_{1 \le j \le n} \bigwedge_{0 < i < k \le n} \neg(p_{ij} \wedge p_{kj})$$

# N-Queens in SAT

- Constraints

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_{11}$ | $p_{12}$ | $p_{13}$ | $p_{14}$ | $p_{15}$ | $p_{16}$ | $p_{17}$ | $p_{18}$ |
| $p_{21}$ | $p_{22}$ | $p_{23}$ | $p_{24}$ | $p_{25}$ | $p_{26}$ | $p_{27}$ | $p_{28}$ |
| $p_{31}$ | $p_{32}$ | $p_{33}$ | $p_{34}$ | $p_{35}$ | $p_{36}$ | $p_{37}$ | $p_{38}$ |
| $p_{41}$ | $p_{42}$ | $p_{43}$ | $p_{44}$ | $p_{45}$ | $p_{46}$ | $p_{47}$ | $p_{48}$ |
| $p_{51}$ | $p_{52}$ | $p_{53}$ | $p_{54}$ | $p_{55}$ | $p_{56}$ | $p_{57}$ | $p_{58}$ |
| $p_{61}$ | $p_{62}$ | $p_{63}$ | $p_{64}$ | $p_{65}$ | $p_{66}$ | $p_{67}$ | $p_{68}$ |
| $p_{71}$ | $p_{72}$ | $p_{73}$ | $p_{74}$ | $p_{75}$ | $p_{76}$ | $p_{77}$ | $p_{78}$ |
| $p_{81}$ | $p_{82}$ | $p_{83}$ | $p_{84}$ | $p_{85}$ | $p_{86}$ | $p_{87}$ | $p_{88}$ |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_{11}$ | $p_{12}$ | $p_{13}$ | $p_{14}$ | $p_{15}$ | $p_{16}$ | $p_{17}$ | $p_{18}$ |
| $p_{21}$ | $p_{22}$ | $p_{23}$ | $p_{24}$ | $p_{25}$ | $p_{26}$ | $p_{27}$ | $p_{28}$ |
| $p_{31}$ | $p_{32}$ | $p_{33}$ | $p_{34}$ | $p_{35}$ | $p_{36}$ | $p_{37}$ | $p_{38}$ |
| $p_{41}$ | $p_{42}$ | $p_{43}$ | $p_{44}$ | $p_{45}$ | $p_{46}$ | $p_{47}$ | $p_{48}$ |
| $p_{51}$ | $p_{52}$ | $p_{53}$ | $p_{54}$ | $p_{55}$ | $p_{56}$ | $p_{57}$ | $p_{58}$ |
| $p_{61}$ | $p_{62}$ | $p_{63}$ | $p_{64}$ | $p_{65}$ | $p_{66}$ | $p_{67}$ | $p_{68}$ |
| $p_{71}$ | $p_{72}$ | $p_{73}$ | $p_{74}$ | $p_{75}$ | $p_{76}$ | $p_{77}$ | $p_{78}$ |
| $p_{81}$ | $p_{82}$ | $p_{83}$ | $p_{84}$ | $p_{85}$ | $p_{86}$ | $p_{87}$ | $p_{88}$ |

  - At most one queen on each diagonal in both directions.

# N-Queens in SAT

- Observation
  - $p_{ij}$ and $p_{i'j'}$ are on the same diagonal iff

$$i + j = i' + j' \text{ or } i - j = i' - j'$$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_{11}$ | $p_{12}$ | $p_{13}$ | $p_{14}$ | $p_{15}$ | $p_{16}$ | $p_{17}$ | $p_{18}$ |
| $p_{21}$ | $p_{22}$ | $p_{23}$ | $p_{24}$ | $p_{25}$ | $p_{26}$ | $p_{27}$ | $p_{28}$ |
| $p_{31}$ | $p_{32}$ | $p_{33}$ | $p_{34}$ | $p_{35}$ | $p_{36}$ | $p_{37}$ | $p_{38}$ |
| $p_{41}$ | $p_{42}$ | $p_{43}$ | $p_{44}$ | $p_{45}$ | $p_{46}$ | $p_{47}$ | $p_{48}$ |
| $p_{51}$ | $p_{52}$ | $p_{53}$ | $p_{54}$ | $p_{55}$ | $p_{56}$ | $p_{57}$ | $p_{58}$ |
| $p_{61}$ | $p_{62}$ | $p_{63}$ | $p_{64}$ | $p_{65}$ | $p_{66}$ | $p_{67}$ | $p_{68}$ |
| $p_{71}$ | $p_{72}$ | $p_{73}$ | $p_{74}$ | $p_{75}$ | $p_{76}$ | $p_{77}$ | $p_{78}$ |
| $p_{81}$ | $p_{82}$ | $p_{83}$ | $p_{84}$ | $p_{85}$ | $p_{86}$ | $p_{87}$ | $p_{88}$ |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $p_{11}$ | $p_{12}$ | $p_{13}$ | $p_{14}$ | $p_{15}$ | $p_{16}$ | $p_{17}$ | $p_{18}$ |
| $p_{21}$ | $p_{22}$ | $p_{23}$ | $p_{24}$ | $p_{25}$ | $p_{26}$ | $p_{27}$ | $p_{28}$ |
| $p_{31}$ | $p_{32}$ | $p_{33}$ | $p_{34}$ | $p_{35}$ | $p_{36}$ | $p_{37}$ | $p_{38}$ |
| $p_{41}$ | $p_{42}$ | $p_{43}$ | $p_{44}$ | $p_{45}$ | $p_{46}$ | $p_{47}$ | $p_{48}$ |
| $p_{51}$ | $p_{52}$ | $p_{53}$ | $p_{54}$ | $p_{55}$ | $p_{56}$ | $p_{57}$ | $p_{58}$ |
| $p_{61}$ | $p_{62}$ | $p_{63}$ | $p_{64}$ | $p_{65}$ | $p_{66}$ | $p_{67}$ | $p_{68}$ |
| $p_{71}$ | $p_{72}$ | $p_{73}$ | $p_{74}$ | $p_{75}$ | $p_{76}$ | $p_{77}$ | $p_{78}$ |
| $p_{81}$ | $p_{82}$ | $p_{83}$ | $p_{84}$ | $p_{85}$ | $p_{86}$ | $p_{87}$ | $p_{88}$ |

# N-Queens in SAT

- At most one queen on each diagonal

$$\bigwedge_{1 \le i < i' \le n} \; \bigwedge_{j,j' : i+j=i'+j' \,\vee\, i-j=i'-j'} \neg(p_{ij} \wedge p_{i'j'})$$

# N-Queens in SAT

- How to find all solutions?
  - After finding a solution, add the negation of it to the formula, and repeat this until the formula is UNSAT.

# Homework: Sudoku in SAT

- Variables

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 9 | 8 | 5 | 6 | | | |
| | 8 | | | | 9 | | | |
| 2 | | | | | 7 | | | |
| 7 | | | | | 1 | 3 | 9 | 6 |
| 9 | | | | 6 | | | | 5 |
| 5 | 3 | 6 | 2 | | | | | 7 |
| | | | 9 | | | | | 1 |
| | | | 3 | | | | 6 | |
| | | | 6 | 8 | 2 | 4 | | |

  - A Boolean variable $p_{ijk}$ for each possible value $\{1, \ldots, 9\}$ of each $9$x$9$ position on the board.
    - $p_{ijk} = T$ means there is $k$ on row $i$ and column $j$.

# Arithmetics in SAT

- 7 + 21 in SAT?

# Addition in Propositional Logic

- 7+21 in SAT?
  - Represent each number in base 2 via $n$ Binary variables taking values $T = 1$ and $F = 0$.
  - $a_{n-1} \ldots a_0$ corresponds to a number $a$ in base 10 as:

$$a = \sum_{i=0}^{n-1} a_i * 2^i \qquad a_i \in \{0,1\}$$

  - E.g., 01101 represents 1 + 4 + 8 = 13.

# Addition in Propositional Logic

- Decision problem
  - Given $a$ and $b$ (represented in binary), find $d$ (represented in binary) satisfying $a + b = d$.
- Variables
  - $a_{n-1} \dots a_0, b_{n-1} \dots b_0, d_{n-1} \dots d_0$
  - Carries $c_n c_{n-1} \dots c_0$
    - 0 + 0 + 0 = 0, carry = 0
    - 0 + 0 + 1 = 1, carry = 0
    - 0 + 1 + 1 = 0, carry = 1
    - 1 + 1 + 1 = 1, carry = 1

# Addition in Propositional Logic

- Decision problem
  - Given $a$ and $b$ (represented in binary), find $d$ (represented in binary) satisfying $a + b = d$.
- Variables
  - $a_{n-1} \ldots a_0, b_{n-1} \ldots b_0, d_{n-1} \ldots d_0$
  - Carries $c_n c_{n-1} \ldots c_0$
- Constraints
  - Compute $d_i$ from right to left starting from $c_0 = 0$.

# Example

$$
\begin{array}{l|cccccc}
c \rightarrow & 0 & 0 & 1 & 1 & 1 & 0 \\
a = 7 \rightarrow & & 0 & 0 & 1 & 1 & 1 \\
b = 21 \rightarrow & & 1 & 0 & 1 & 0 & 1 \\
\hline
d = 28 \rightarrow & & 1 & 1 & 1 & 0 & 0
\end{array}
$$

# Addition in Propositional Logic

- $d_i = a_i + \ b_i + c_i \quad mod\ 2, \qquad i = 0, \ldots, n-1$

  $$a_i \leftrightarrow b_i \leftrightarrow c_i \leftrightarrow d_i$$

- $c_{i+1} = 1 \ \leftrightarrow \ a_i + b_i + c_i > 1, \ \ i = 0, \ldots, n-1$

  $$c_{i+1} \leftrightarrow (a_i \wedge b_i) \vee (a_i \wedge c_i) \vee (b_i \wedge c_i)$$

- $c_n = 0$ (to fit in $n$ bits) and $c_0 = 0$ (initial carry)

  $$\neg c_n \wedge \neg c_0$$

# Addition in Propositional Logic

- Let $f$ be the conjunction of all these formulas.
- To compute $a + b = d$, the final formula is:

$$f \wedge \bigwedge_{i=0}^{n-1} [\neg] a_i \wedge \bigwedge_{i=0}^{n-1} [\neg] b_i$$

- E.g., when $a = 13,\ b = 7$, we have:

$$f \ \wedge \underbrace{\neg a_1 \wedge a_2 \wedge a_3 \wedge \neg a_4 \wedge a_5}_{a\,=\,13\,=\,01101} \wedge \underbrace{\neg b_1 \wedge \neg b_2 \wedge b_3 \wedge b_4 \wedge b_5}_{b\,=\,7\,=\,00111}$$

for which a SAT solver would return the result:

$$\underbrace{d_1 \wedge \neg d_2 \wedge d_3 \wedge \neg d_4 \wedge \neg d_5}_{d\,=\,20\,=\,10100}$$

# Addition in Propositional Logic

- If $d$ does not fit in $n$ digits, $c_n$ will be forced to be 1, and the result will be UNSAT.
  - Solution: expand the formula with leading bits in $a$ and $b$ set to zero.

$$c_n = 1,\ a_n = 0,\ b_n = 0$$

# Subtraction in Propositional Logic

- $a + b = d \leftrightarrow b = d - a$
- Compute $b$ using the previous $f$.

$$f \wedge \bigwedge_{i=0}^{n-1} [\neg] a_i \wedge \bigwedge_{i=0}^{n-1} [\neg] d_i$$

# Homework: Multiplication in SAT

- $mul(a, b, r)$ which encodes $a * b = r$

# Arithmetic in Propositional Logic

- Multiplication for factorizing a number
  - $f(r) = mul(a, b, r) \wedge a > 1 \wedge b > 1$
  - How can we express $a > 1 \wedge b > 1$ ?

    $(a_{n-1} \vee a_{n-2} \vee \ldots a_0) \wedge (b_{n-1} \vee b_{n-2} \vee \ldots b_0)$
- Prime number
  - $r$ is a prime number iff $f(r)$ is UNSAT.
  - E.g., $f(1234567891)$ is proved UNSAT, while $f(1234567897)$ is proved SAT.