УНИВЕРСИТЕТ ИТМО

# Решение задач путем сведения к SAT. Практические аспекты использования SAT-решателей.

Чухарев Константин // Университет ИТМО

Сириус
Образовательный центр

Санкт-Петербург, 2020

ITMO UNIVERSITY
Saint Petersburg, Russia

# Problem Solving Using SAT Solvers

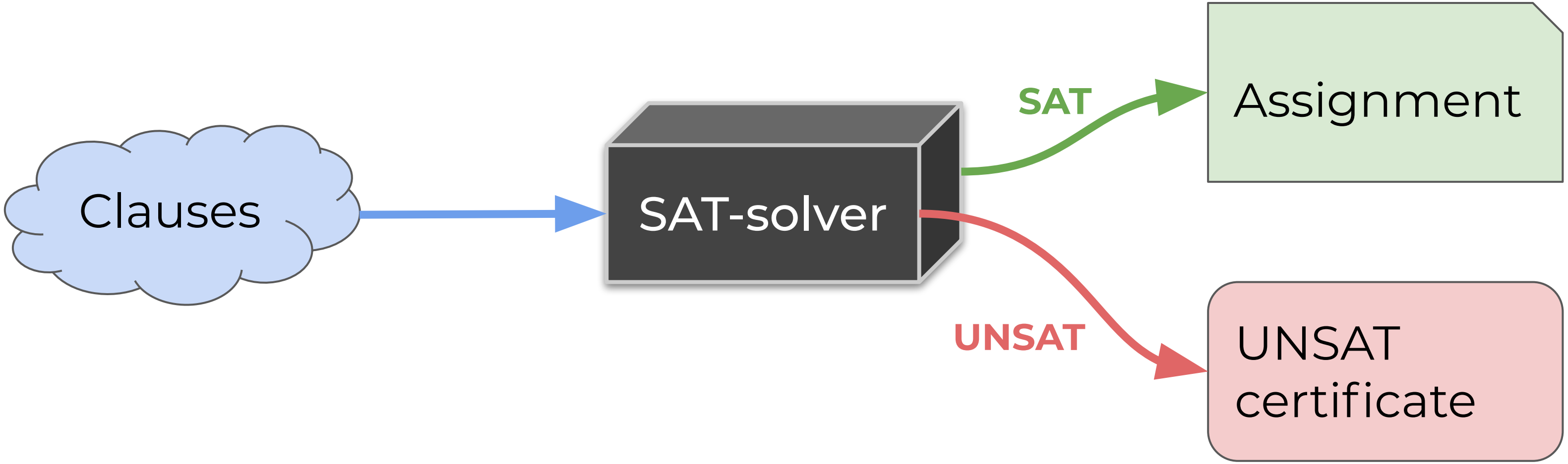
Chukharev Konstantin // ITMO University

This logo should also be in English...

Сириус
Образовательный центр

Saint Petersburg, 2020

# Solving via SAT Solvers



# Definitions [1/2]

**Boolean variables:** `x, y, z`

**Literal** – variable or its negation: `x, ~x`

**Propositional connectives:** $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\Leftrightarrow$

**Propositional (Boolean) formula grammar:**

$$F ::= L \mid {\sim}F \mid F \wedge F \mid F \vee F \mid F \rightarrow F \mid F \Leftrightarrow F$$

// F – Boolean formula, L – Literal

# Definitions [2/2]

**Assignment:** $\nu : X \to \{0, u, 1\} \qquad 0 < u < 1$

Clause (of CNF) is:

*satisfied*, if *at least one* literal is assigned 1,
*unsatisfied*, if *all* its literals are assigned 0,
*unresolved*, otherwise.

In case you forgot what clause looks like:

$$x \vee y \vee z$$

CNF formula φ is:

*satisfied*, if *all* of its clauses are satisfied,
*unsatisfied*, if *at least one* clause is unsatisfied,
unresolved, otherwise.

SAT problem

ITMOre than a UNIVERSITY

# Normal Forms

$F ::= L \mid \sim F \mid F \land F \mid F \lor F \mid F \to F \mid F \Leftrightarrow F$

**desugar:**

$F ::= L \mid \sim F \mid F \land F \mid F \lor F$

**NNF:**

$F ::= L \mid F \land F \mid F \lor F$

**DNF:**

$F :: C \mid C \lor F$ $\quad$ $C ::= L \mid L \land C$ $\quad$ // C – $\land$-clause

**CNF:**

$F :: D \mid D \land F$ $\quad$ $D ::= L \mid L \lor D$ $\quad$ // D – $\lor$-clause

# Naïve Conversion to CNF

↳ Exponential blow up:

$$n \left\{ \begin{array}{l} (x_1 \wedge y_1) \vee \\ (x_2 \wedge y_2) \vee \\ \quad \cdots \\ (x_n \wedge y_n) \end{array} \right. \xrightarrow{\text{CNF}} \left. \begin{array}{l} (x_1 \vee x_2 \vee \cdots \vee x_n) \wedge \\ (y_1 \vee x_2 \vee \cdots \vee x_n) \wedge \\ \quad \cdots \\ (y_1 \vee y_2 \vee \cdots \vee y_n) \end{array} \right\} 2^n$$

# Tseytin Transformation

↳ Conversion of arbitrary Boolean formula to **equisatisfiable** CNF.

$$t \equiv A \wedge B \quad \xrightarrow{\text{Tseytin}} \quad (\overline{A} \vee \overline{B} \vee t) \wedge (A \vee \overline{t}) \wedge (B \vee \overline{t})$$

$$t \equiv A \vee B \quad \xrightarrow{\text{Tseytin}} \quad (A \vee B \vee \overline{t}) \wedge (\overline{A} \vee t) \wedge (\overline{B} \vee t)$$

// *t* – auxiliary variable

**More:** https://en.wikipedia.org/wiki/Tseytin_transformation

**Note:** the resulting CNF is **not equivalent** to the original formula!

# Problem Representation

↳ Declarative description of the sought solution.

**Declarative** = **variables** and **constraints**.
Variables are finite-domain.
Constraints are CNF clauses (discussed earlier).

**Issue:** Modeling with *finite-domain variables*, but limited to use only *propositional* ones in SAT. **What to do?**

# Onehot Encoding

↳ a.k.a. *"sparse"*, *"direct"*, *"pairwise"*

$$X \in [1..n] \bowtie_{\text{onehot}} \{x_1, \dots, x_n\}$$

$$x_1 \Leftrightarrow (X = 1) \qquad x_2 \Leftrightarrow (X = 2) \qquad \dots \qquad x_n \Leftrightarrow (X = n)$$

$$\text{AtLeastOne}(x_1, \dots, x_n) \wedge \text{AtMostOne}(x_1, \dots, x_n)$$

$$\text{AtLeastOne}(x_1, \dots, x_n) \equiv \bigvee_{1 \le i \le n} x_i$$

$$\text{AtMostOne}(x_1, \dots, x_n) \equiv \bigwedge_{1 \le i < j \le n} (x_i \implies \neg x_j)$$

# Cardinality Constraints

$$k_{\min} \le \sum_{1 \le i \le n} x_i \le k_{\max}$$

Encode in CNF using one of the following techniques:

- **Totalizer:** Bailleux O., Boufkhad Y. Efficient CNF Encoding of Boolean Cardinality Constraints, 2003. DOI: 10.1007/978-3-540-45193-8_8.
- **Sequential counters:** Sinz, C. Towards an optimal CNF encoding of Boolean cardinality constraints, 2005. DOI: https://doi.org/10.1007/11564751_73.
- **Sorting networks:** Een, N., Sörensson, N. Translating Pseudo-Boolean Constraints into SAT, 2005. DOI: https://doi.org/10.3233/sat190014.
- Discover anything even better.

# Searching for SAT

If your problem is parametrized, and you want to find the most optimal value of the parameter, then you have two options:

- ↳ Bottom-up search (UNSAT ⟶ UNSAT ⟶ … ⟶ SAT!)
- ↳ Upside-down search (SAT ⟶ SAT⟶ … ⟶ SAT! ⟶ UNSAT)

# Searching for Multiple Solutions

If you want to find multiple (all) solutions, just do the following:

1. Find any solution
2. Ban it
3. Repeat until UNSAT

# Demo time

# Contest

https://sirius2020.contest.codeforces.com

...

Thanks for your attention.

kchukharev@itmo.ru

Lipen

ITsMOre than a UNIVERSITY