

# Proof Complexity and Solving LAB

Winter Semester 2020

Dr. Joshua Blinkhorn

Friedrich-Schiller-Universität Jena

<https://github.com/JoshuaBlinkhorn/SAT-LAB>

# Goals

- Implementation of SAT solving algorithms
  - (a) 2-SAT (polynomial time)
  - (b) DPLL (decision tree)
  - (c) CDCL
    - clause learning
    - watched literals
    - decision heuristics
    - restart strategy
  - (d) QBF expansion..
- Practical programming experience
  - use your favourite language (Python, C, C++, Java, ..)
  - recommended: **Python**

# Conjunctive Normal Form (CNF)

- a literal is a variable or its negation
- a clause is a disjunction of literals
- a CNF is a conjunction of clauses

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

- the **width** of a clause is the number of literals
- a **tautological** clause contains a variable and its negation, e.g.:

$$(x_1 \vee \neg x_1) \quad \text{or} \quad (\neg x_2 \vee \neg x_3 \vee x_2)$$

# The DIMACS CNF Encoding

- machine readable encoding
- variables are natural numbers:  $x_1 \mapsto 1$ ,  $x_2 \mapsto 2$  etc.
- negation represented by minus:  $\neg x_1 \mapsto -1$ ,  $\neg x_2 \mapsto -2$  etc.

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

```
p cnf 3 4
1 2 3 0
-1 -2 0
-3 0
-1 3 0
```

# The DIMACS CNF Encoding

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

```
p cnf 3 4
1 2 3 0
-1 -2 0
-3 0
-1 3 0
```

- begins with the problem line

p cnf *[number of variables]* *[number of clauses]*

- other lines represent clauses as zero-terminated literal sets

## First Task – Random CNF Generator

- A random  $(n, c, k)$ -CNF is:
  - a random CNF with  $n$  variables and  $c$  clauses
  - all clauses of width  $k$
  - no tautological clauses
  - no repeated clauses
  - no repeated literals in clauses
  - clauses selected uniformly at random
- Implement a tool:

`random-cnf t n c k`

which outputs  $t$  random  $(n, c, k)$ -CNF formulas in DIMACS

## Second Task – Testing

- Install a SAT solver

<http://fmv.jku.at/lingeling/>

- Test your random CNFs with the solver
- Can you find out anything interesting by varying the parameters  $n$ ,  $c$  and  $k$ ?