Introduction to MiniSAT

Spring 2019
CSCE 235H Introduction to Discrete Structures
URL: cse.unl.edu/~cse235h
All questions: Piazza

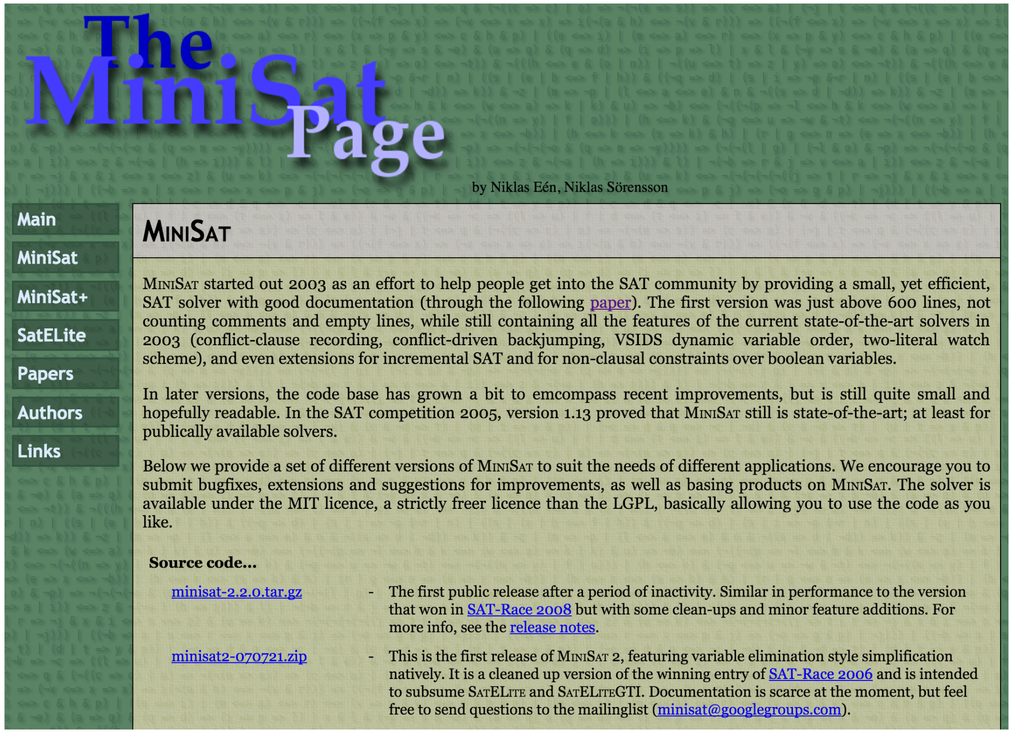
The MiniSAT Solver

- ‘Minimalistic,’ open-source SAT solver
- Designed to be easy to learn and modify
- Highly performant: has won SAT competitions
- Used as the backend for many projects

Setting up MiniSAT (1)

- Access the MiniSAT webpage at

  http://minisat.se/MiniSat.html



by Niklas Eén, Niklas Sörensson

MINISAT

MINISAT started out 2003 as an effort to help people get into the SAT community by providing a small, yet efficient, SAT solver with good documentation (through the following paper). The first version was just above 600 lines, not counting comments and empty lines, while still containing all the features of the current state-of-the-art solvers in 2003 (conflict-clause recording, conflict-driven backjumping, VSIDS dynamic variable order, two-literal watch scheme), and even extensions for incremental SAT and for non-clausal constraints over boolean variables.

In later versions, the code base has grown a bit to emcompass recent improvements, but is still quite small and hopefully readable. In the SAT competition 2005, version 1.13 proved that MINISAT still is state-of-the-art; at least for publically available solvers.

Below we provide a set of different versions of MINISAT to suit the needs of different applications. We encourage you to submit bugfixes, extensions and suggestions for improvements, as well as basing products on MINISAT. The solver is available under the MIT licence, a strictly freer licence than the LGPL, basically allowing you to use the code as you like.

Source code...

minisat-2.2.0.tar.gz	-	The first public release after a period of inactivity. Similar in performance to the version that won in SAT-Race 2008 but with some clean-ups and minor feature additions. For more info, see the release notes.
minisat2-070721.zip	-	This is the first release of MINISAT 2, featuring variable elimination style simplification natively. It is a cleaned up version of the winning entry of SAT-Race 2006 and is intended to subsume SATELITE and SATELITEGTI. Documentation is scarce at the moment, but feel free to send questions to the mailinglist (minisat@googlegroups.com).

Setting up MiniSAT (2)

- Download the file `minisat-2.2.0.tar.gz`
- Copy the file to the CSE server

OR

- Login to the CSE server
- Download the file directly

```
wget
http://minisat.se/downloads/minisat-
2.2.0.tar.gz
```

Setting up MiniSAT (3)

- Untar the MiniSAT files

```
tar -zxvf minisat-2.2.0.tar.gz
```

- Enter the minisat directory

```
cd minisat
```

Setting up MiniSAT (4)

- Use the pwd command to "print working directory"

```
pwd
```

- Set the MROOT environment variable to the full path to the minisat directory (only required for compilation)

```
setenv MROOT <full path to minisat>
```

Setting up MiniSAT (5)

- Enter the core directory

```
cd core
```

- Compile the program using make

```
make
```

- The minisat executable is now located in the `core` directory

Using MiniSAT

- Command usage

```
./minisat [options] <input-file>
        <result-output-file>
```

- See full listing of options

```
./minisat --help
```

DIMACS CNF format (1)

- Created by the *Center for Discrete Mathematics and Theoretical Computer Science* (DIMACS)
- Standard format for expressing CNF formulas
- Human-readable text file

DIMACS CNF format (2)

Comments start with a ‘c’

Each line expresses one clause and must end with a ‘0’

```
c
c start with comments
c
c
p cnf 5 3
1 -5 4 0
-1 5 3 4 0
-3 -4 0
```

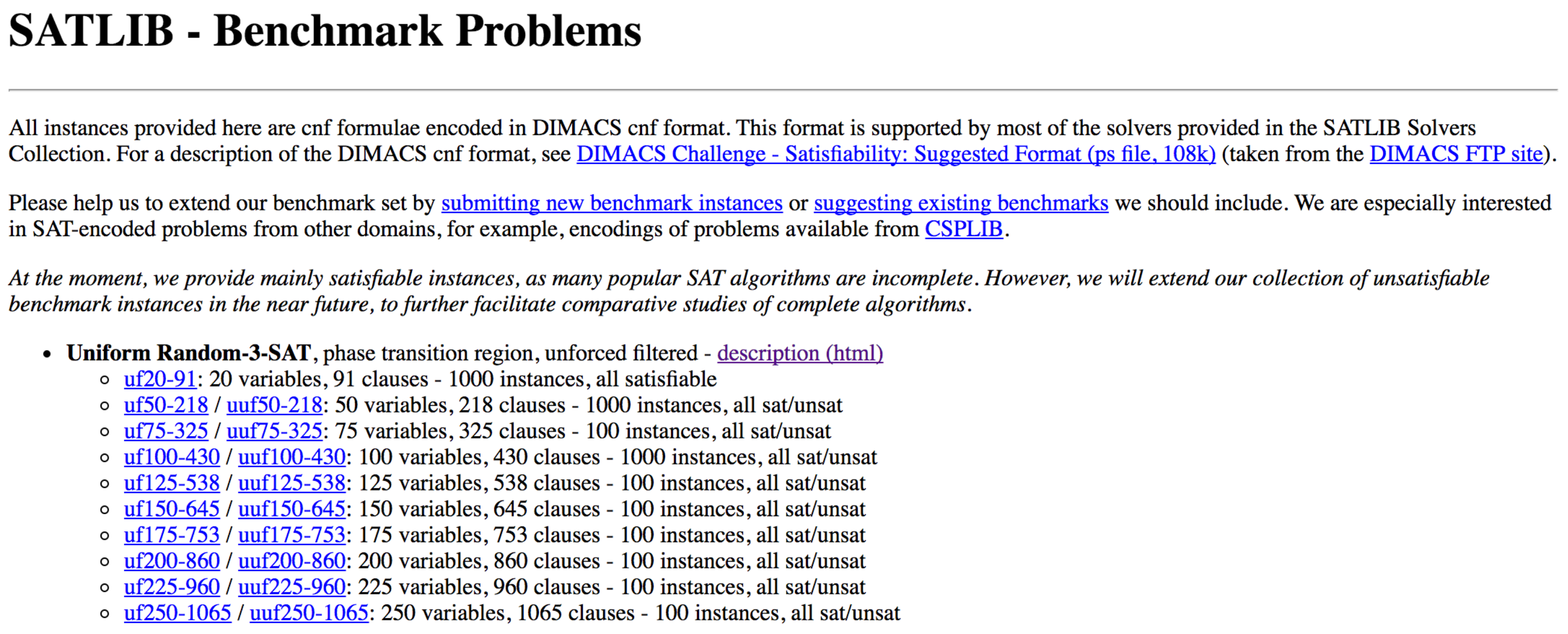
Parameters line starts with a ‘p’ and indicates cnf format, 5 variables, 3 clauses

Variables must be non-zero numbers. Negated variables have a minus sign

SATLIB benchmark problems (1)

- Access the SATLIB webpage at

  http://www.cs.ubc.ca/~hoos/SATLIB/benchm.html



SATLIB - Benchmark Problems

All instances provided here are cnf formulae encoded in DIMACS cnf format. This format is supported by most of the solvers provided in the SATLIB Solvers Collection. For a description of the DIMACS cnf format, see DIMACS Challenge - Satisfiability: Suggested Format (ps file, 108k) (taken from the DIMACS FTP site).

Please help us to extend our benchmark set by submitting new benchmark instances or suggesting existing benchmarks we should include. We are especially interested in SAT-encoded problems from other domains, for example, encodings of problems available from CSPLIB.

At the moment, we provide mainly satisfiable instances, as many popular SAT algorithms are incomplete. However, we will extend our collection of unsatisfiable benchmark instances in the near future, to further facilitate comparative studies of complete algorithms.

- **Uniform Random-3-SAT**, phase transition region, unforced filtered - description (html)
  - uf20-91: 20 variables, 91 clauses - 1000 instances, all satisfiable
  - uf50-218 / uuf50-218: 50 variables, 218 clauses - 1000 instances, all sat/unsat
  - uf75-325 / uuf75-325: 75 variables, 325 clauses - 100 instances, all sat/unsat
  - uf100-430 / uuf100-430: 100 variables, 430 clauses - 1000 instances, all sat/unsat
  - uf125-538 / uuf125-538: 125 variables, 538 clauses - 100 instances, all sat/unsat
  - uf150-645 / uuf150-645: 150 variables, 645 clauses - 100 instances, all sat/unsat
  - uf175-753 / uuf175-753: 175 variables, 753 clauses - 100 instances, all sat/unsat
  - uf200-860 / uuf200-860: 200 variables, 860 clauses - 100 instances, all sat/unsat
  - uf225-960 / uuf225-960: 225 variables, 960 clauses - 100 instances, all sat/unsat
  - uf250-1065 / uuf250-1065: 250 variables, 1065 clauses - 100 instances, all sat/unsat

SATLIB benchmark problems (2)

- Repository of SAT benchmark problems
- Used the evaluate and compare solvers
- Taken from a variety of domains
  - E.g., randomly generated, puzzles, coloring graph, planning, etc.
  - Descriptions of the benchmarks are included

SATLIB benchmark problems (3)

- ALERT
  - Some instances may have extra lines that cannot be parsed by MiniSAT.
  - Delete any line starting with ‘%’ and all following lines.