Introduction to MiniSAT

Spring 2018

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



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

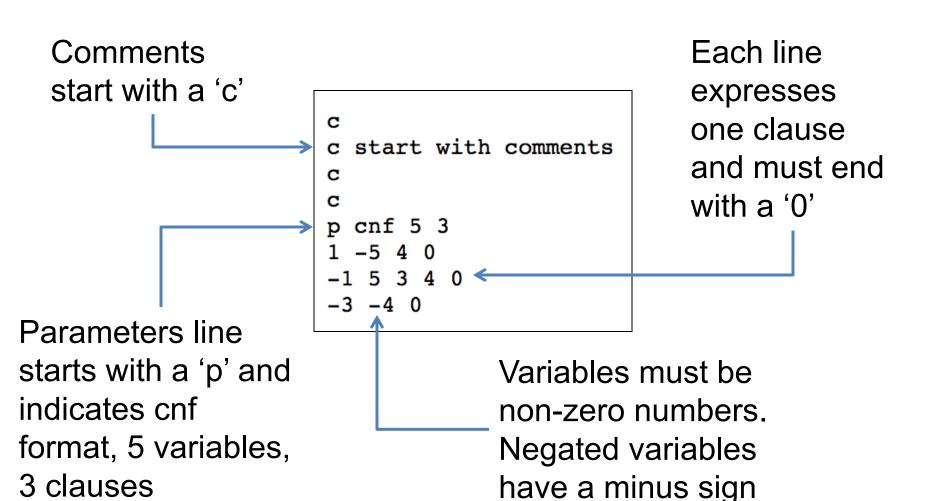
See full listing of options

```
./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)



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 <u>DIMACS Challenge - Satisfiability: Suggested Format (ps file, 108k)</u> (taken from the <u>DIMACS FTP site</u>).

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

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)
 - o uf20-91: 20 variables, 91 clauses 1000 instances, all satisfiable
 - o <u>uf50-218</u> / <u>uuf50-218</u>: 50 variables, 218 clauses 1000 instances, all sat/unsat
 - o uf75-325 / uuf75-325: 75 variables, 325 clauses 100 instances, all sat/unsat
 - o <u>uf100-430</u> / <u>uuf100-430</u>: 100 variables, 430 clauses 1000 instances, all sat/unsat
 - o uf125-538 / uuf125-538: 125 variables, 538 clauses 100 instances, all sat/unsat
 - <u>uf150-645</u> / <u>uuf150-645</u>: 150 variables, 645 clauses 100 instances, all sat/unsat
 - uf175-753 / uuf175-753: 175 variables, 753 clauses 100 instances, all sat/unsat
 - o <u>uf200-860</u> / <u>uuf200-860</u>: 200 variables, 860 clauses 100 instances, all sat/unsat
 - o uf225-960 / uuf225-960: 225 variables, 960 clauses 100 instances, all sat/unsat
 - o <u>uf250-1065</u> / <u>uuf250-1065</u>: 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.