The SAT4J library, release 2.2

How SAT4J MAXSAT really works

Daniel Le Berre et Anne Parrain

CRIL-CNRS UMR 8188 - Université d'Artois

Pragmatics of SAT workshop, FLoC , July 10, 2010, Edinburgh

Outline

A generic and flexible SAT solver

To build Pseudo-Boolean solvers

That can solve Optimization problems

An insight of a new solver submitted to PB 2010

Solving more problems : MAXSAT, WBO, etc.

Conclusion

A generic and flexible CDCL solver

Basis Minisat 1.10 specification + conflict minimization from Minisat 1.13

Static Restarts strategies Minisat, Biere, Luby

Generic Conflict minimization None, Simple, Expensive
works with all constraints and data structures

Learning LimitedLearning, LearnAllClauses, NoLearning, ...
learning is not coupled with conflict analysis

Learned clauses deletion Memory based, Glucose

Phase selection Random, Positive, Negative, AppearInLastLearnedClauses, RSAT phase caching

Lazy Data structures Watched Literals, Head/Tail

Default configuration

Outline

A generic and flexible SAT solver

To build Pseudo-Boolean solvers

That can solve Optimization problems

An insight of a new solver submitted to PB 2010

Solving more problems : MAXSAT, WBO, etc.

Conclusion

Linear Pseudo-Boolean decision and optimization problems

Linear Pseudo-Boolean constraint

$$-3x_1 + 4x_2 - 7x_3 + x_4 \leq -5$$

- variables x_i take their value in $\{0, 1\}$
- $\overline{x_1} = 1 - x_1$
- coefficients and degree are integer-valued constants

Pseudo-Boolean decision problem : NP-complete

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \\ (c) & x_1 + \overline{x_2} + x_5 \geq 1 \end{cases}$$

Plus an objective function : Optimization problem, NP-hard

$$min : 4x_2 + 2x_3 + x_5$$

Rules on LPB constraints : Linear combination, division

linear combination:

$$\frac{\sum_i a_i.x_i \geq k \qquad \sum_i a'_i.x_i \geq k'}{\sum_i (\alpha.a_i + \alpha'.a'_i).x_i \geq \alpha.k + \alpha'.k'}$$

with $\alpha > 0$ and $\alpha' > 0$

$$\frac{x_1 + x_2 + 3x_3 + x_4 \geq 3 \qquad 2\overline{x_1} + 2\overline{x_2} + x_4 \geq 3}{2x_1 + 2x_2 + 6x_3 + 2x_4 + 2\overline{x_1} + 2\overline{x_2} + x_4 \geq 2 \times 3 + 3}$$

$$2x_1 + 2x_2 + 6x_3 + 2x_4 + 2 - 2x_1 + 2 - 2x_2 + x_4 \geq 9$$

$$6x_3 + 3x_4 \geq 5$$

Note that $2x + 2\overline{x} = 2$, not 0, the coefficients are growing !

division :

$$\frac{5x_3 + 3x_4 \geq 5}{\lceil 5/5 \rceil x_3 + \lceil 3/5 \rceil x_4 \geq \lceil 5/5 \rceil}$$

$$x_3 + x_4 \geq 1$$

One can always reduce a LPB constraint to a clause !

Cutting planes proof system

- Linear combination + division = cutting plane proof system (complete).
- First introduced for linear programming by R. Gomory in 1958
- Cutting planes can be seen as a generalization of the resolution (J.N. Hooker, 1988)
- Resolution is used during conflict analysis in CDCL solvers → just replace Resolution by Cutting Planes during Conflict Analysis to build a new solver with a better proof system ! (Done since PBChaff and Galena solvers, 2003)

Pseudo Boolean solvers in SAT4J

SAT4J PB RES Generic SAT solver with resolution inference during conflict analysis (learn clauses) with input constraints allowed to be Pseudo-Boolean constraints

SAT4J PB CuttingPlanes Generic SAT solver with cutting planes based inference during conflict analysis (learn PB constraints) with input constraints allowed to be Pseudo-Boolean constraints

- Resolution based PB solver takes advantage of the full existing SAT machinery (like PBS, SATZOO, Minisat 1.10, ...)
- Cutting Planes based PB solver need to deal with new data structures and algorithms : no lazy data structure for constraints, need arbitrary precision arithmetic for correctness (unlike PBChaff, Galena, Pueblo, ...)

Why two PB engines ?

- The resolution based PB solver is usually faster than the CP based one
- Some benchmarks can only be solved using CP solver (e.g. pigeon hole)
- For a specific application, it is better to try both (aggregation order example)
- The principles behind each solver are clear : no tweaks to solve a few more benchmarks during the PB evaluations !

Outline

A generic and flexible SAT solver

To build Pseudo-Boolean solvers

That can solve Optimization problems

An insight of a new solver submitted to PB 2010

Solving more problems : MAXSAT, WBO, etc.

Conclusion

Optimization using strengthening (linear search)

input : A set of clauses, cardinalities and pseudo-boolean constraints setOfConstraints and an objective function objFct to minimize

output: a model of setOfConstraints, or UNSAT if the problem is unsatisfiable.

```
answer ← isSatisfiable (setOfConstraints);
if answer is UNSAT then
    return UNSAT
end
repeat
    model ← answer ;
    answer ← isSatisfiable (setOfConstraints ∪
                              {objFct < objFct (model)});
until (answer is UNSAT);
return model ;
```

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Objective function

$$\min : \quad 4x_2 + 2x_3 + x_5$$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Model

$\overline{x_1}, x_2, \overline{x_3}, x_4, x_5$

Objective function

min : $4x_2 + 2x_3 + x_5$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Model

$\overline{x_1}, x_2, \overline{x_3}, x_4, x_5$

Objective function

min : $4x_2 + 2x_3 + x_5$

Objective function value

$<$ 5

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Objective function

min : $4x_2 + 2x_3 + x_5$ $<$ 5

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Model

$x_1, \overline{x_2}, x_3, \overline{x_4}, x_5$

Objective function

min : $4x_2 + 2x_3 + x_5 \quad < \quad 5$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Model

$x_1, \overline{x_2}, x_3, \overline{x_4}, x_5$

Objective function

min : $4x_2 + 2x_3 + x_5$

$<$

Objective function value

$3 < 5$

UNIVERSITÉ D'ARTOIS

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Objective function

$$\text{min} : \quad 4x_2 + 2x_3 + x_5 \qquad < \qquad 3$$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Model

$x_1, \overline{x_2}, \overline{x_3}, x_4, x_5$

Objective function

min : $4x_2 + 2x_3 + x_5 \quad < \quad 3$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Model

$x_1, \overline{x_2}, \overline{x_3}, x_4, x_5$

Objective function

min : $4x_2 + 2x_3 + x_5$

$<$

Objective function value

$1 < 3$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Objective function

$$\min : \quad 4x_2 + 2x_3 + x_5 \qquad < \qquad 1$$

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Objective function

min : $4x_2 + 2x_3 + x_5 \quad < \quad 1$

Optimization algorithm

Formula :

$$\begin{cases} (a_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (a_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (b) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

Objective function

$$\min : \quad 4x_2 + 2x_3 + x_5 \qquad < \qquad 1$$

The objective function value 1 is optimal for the formula.

$x_1, \overline{x_2}, \overline{x_3}, x_4, x_5$ is an optimal solution.

Outline

A generic and flexible SAT solver

To build Pseudo-Boolean solvers

That can solve Optimization problems

An insight of a new solver submitted to PB 2010

Solving more problems : MAXSAT, WBO, etc.

Conclusion

Remarks about the optimization procedure

- No need for an initial upper bound !
- Phase selection strategy takes into account the objective function.
- External to the PB solver : can use any PB solver.
- SAT, SAT, SAT, ..., SAT, UNSAT pattern
- SAT answer usually easier to provide than UNSAT one
- In practice : optimality is often hard to prove for the Resolution based PB solver (pigeon hole ?).
- Ideally, would like to run the CP PB solver to prove optimality at the end.
- Problem : how to detect that we need to prove optimality ?

Remarks about the optimization procedure

- No need for an initial upper bound !
- Phase selection strategy takes into account the objective function.
- External to the PB solver : can use any PB solver.
- SAT, SAT, SAT, ..., SAT, UNSAT pattern
- SAT answer usually easier to provide than UNSAT one
- In practice : optimality is often hard to prove for the Resolution based PB solver (pigeon hole ?).
- Ideally, would like to run the CP PB solver to prove optimality at the end.
- Problem : how to detect that we need to prove optimality ?
- Nice idea suggested by Olivier Roussel submitted to PB 2010 : run the Res and CP PB solvers in parallel !

Optimization with solvers running in parallel

input : A set of clauses, cardinalities and pseudo-boolean constraints setOfConstraints and an objective function objFct to minimize

output: a model of setOfConstraints, or UNSAT if the problem is unsatisfiable.

```
answer ← isSatisfiable (setOfConstraints);
if answer is UNSAT then
    return UNSAT
end
repeat
    model ← answer ;
    answer ← isSatisfiable (setOfConstraints ∪
                             {objFct < objFct (model)});
until (answer is UNSAT);
return model ;
```

logic-synthesis/normalized-jac3.opb @ PB2010

```
% Cutting Planes
1.17/0.78 c #vars      1731
1.17/0.78 c #constraints  1254
1.76/1.03 c SATISFIABLE
1.76/1.03 c OPTIMIZING...
1.76/1.03 o 26
3.40/1.91 o 25
5.93/3.41 o 24
6.97/4.33 o 23
7.49/4.88 o 22
8.44/5.72 o 21
9.00/6.27 o 20
9.62/6.87 o 19
10.44/7.61 o 18
11.54/8.79 o 17
13.03/10.13 o 16
25.34/22.07 o 15
1800.11/1773.42 s SATISFIABLE
```

```
% Resolution
1.17/0.75 c #vars      1731
1.17/0.75 c #constraints  1254
1.57/0.91 c SATISFIABLE
1.57/0.91 c OPTIMIZING...
1.57/0.91 o 26
2.55/1.42 o 23
2.96/1.60 o 22
3.35/1.80 o 21
16.34/14.32 o 20
55.04/52.91 o 19
766.33/763.00 o 18
1800.04/1795.76 s SATISFIABLE
```

logic-synthesis/normalized-jac3.opb @ PB2010

```
% Cutting Planes
1.17/0.78 c #vars      1731
1.17/0.78 c #constraints  1254
1.76/1.03 c SATISFIABLE
1.76/1.03 c OPTIMIZING...
1.76/1.03 o 26
3.40/1.91 o 25
5.93/3.41 o 24
6.97/4.33 o 23
7.49/4.88 o 22
8.44/5.72 o 21
9.00/6.27 o 20
9.62/6.87 o 19
10.44/7.61 o 18
11.54/8.79 o 17
13.03/10.13 o 16
25.34/22.07 o 15
1800.11/1773.42 s SATISFIABLE
```

```
% Res // CP
1.35/0.84 c #vars      1731
1.35/0.84 c #constraints  1254
1.99/1.85 c SATISFIABLE
1.99/1.85 c OPTIMIZING...
1.99/1.85 o 26 (CuttingPlanes)
2.61/2.89 o 25 (Resolution)
3.91/3.92 o 24 (Resolution)
4.12/5.00 o 23 (Resolution)
5.92/6.01 o 22 (Resolution)
7.72/7.04 o 21 (Resolution)
9.63/8.07 o 20 (CuttingPlanes)
13.04/10.09 o 19 (CuttingPlanes)
15.66/12.10 o 18 (CuttingPlanes)
20.27/15.14 o 17 (CuttingPlanes)
70.03/41.35 o 16 (CuttingPlanes)
218.63/118.14 o 15 (CuttingPlanes)
305.11/164.68 s OPTIMUM FOUND
```

logic-synthesis/normalized-jac3.opb @ PB2010

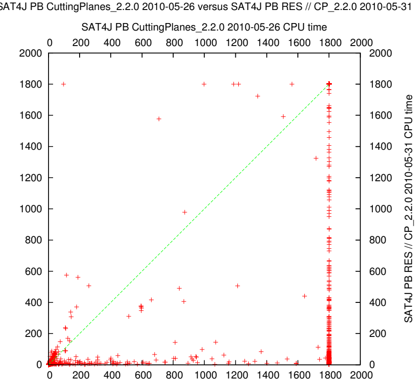
Cutting Planes

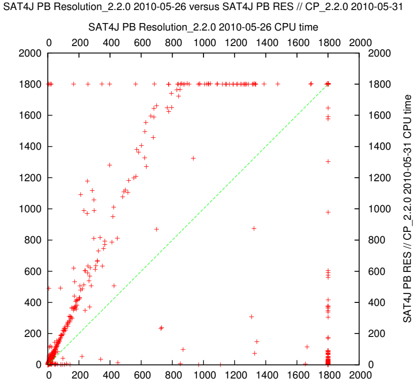
```
1800.11/1773.42 s SATISFIABLE
1800.11/1773.41 c learnt clauses : 2618
1800.11/1773.42 c speed (assignments/second) : 226
```

Res // CP

```
305.11/164.68 s OPTIMUM FOUND
305.11/164.68 c learnt clauses : 1318
305.11/164.68 c speed (assignments/second) : 3927
```

Scatter plots Res // CP vs CP, Resolution





UNIVERSITÉ D'ARTOIS

Regarding the idea to run the two solvers in //

- Res // CP globally better than Res or CP solver during PB 2010 in number of benchmarks solved.
- Res // CP twice as slow as Res on many benchmarks.
- Decision problems : solves the union of the benchmarks solved by Res and CP in half the timeout (CPU time taken into account, not wall clock time).
- Optimization problems : "cooperation" of solvers allow to solve new benchmarks !

Outline

A generic and flexible SAT solver

To build Pseudo-Boolean solvers

That can solve Optimization problems

An insight of a new solver submitted to PB 2010

Solving more problems : MAXSAT, WBO, etc.

Conclusion

Generalized use of selector variables

The minisat+ syndrom : is a SAT solver sufficient for all our needs ?

Selector variable principle : satisfying the selector variable should satisfy the selected constraint.

clause simply add a new variable

$$\bigvee l_i \Rightarrow s \vee \bigvee l_i$$

cardinality add a new weighted variable

$$\sum l_i \geq d \Rightarrow d \times s + \sum l_i \geq d$$

The new constraints is PB, no longer a cardinality !

pseudo add a new weighted variable

$$\sum w_i \times l_i \geq d \Rightarrow d \times s + \sum w_i \times l_i \geq d$$

if the weights are positive, else use

$$(d + \sum_{w_i<0} |w_i|) \times s + \sum w_i \times l_i \geq d$$

From Weighted Partial Max SAT to PBO

Once cardinality constraints, pseudo boolean constraints and objective functions are managed in a solver, one can easily build a weighted partial Max SAT solver

- Add a selector variable s_i per clause C_i : $s_i \vee C_i$
- Objective function : minimize $\sum s_i$
  Reported objective function value is wrong (need to be translated, ok for the Max SAT evaluations)
- Partial MAX SAT : do not add selector variables for hard clauses
- Weighted MAXSAT : use a weighted sum instead of a sum.
  Special case : do not add new variables for unit weighted clauses $w_k l_k$
  Ignore the constraint and add simply $w_k \times \overline{l_k}$ to the objective function.

From WBO to PBO

Weighted Boolean Optimization comes from Weighted CSP
PB constraints can be weighted
The cost of the solution must be strictly smaller than *topcost*
New in PB 2010

- Add a selector variable s_i per weighted constraint
  $[w_i] \sum c_j l_j \geq d$:
  $d \times s_i + \sum c_j l_j \geq d$
- Objective function : minimize $\sum w_i \times s_i$
- Additional constraint : $\sum w_i \times s_i < topcost$

Selector variables + assumptions = explanation

- From the beginning in Minisat 1.12
- Add a new selector variable per constraint
- Check for satisfiability assuming that the selector variables are falsified
- if UNSAT, analyze the final root conflict to keep only selector variables involved in the inconsistency
- Apply a minimization algorithm afterward to compute a minimal explanation (QuickXplain)
- Advantages :
  - not changes needed in the SAT solver internals
  - works for any kind of constraints !

Outline

A generic and flexible SAT solver

To build Pseudo-Boolean solvers

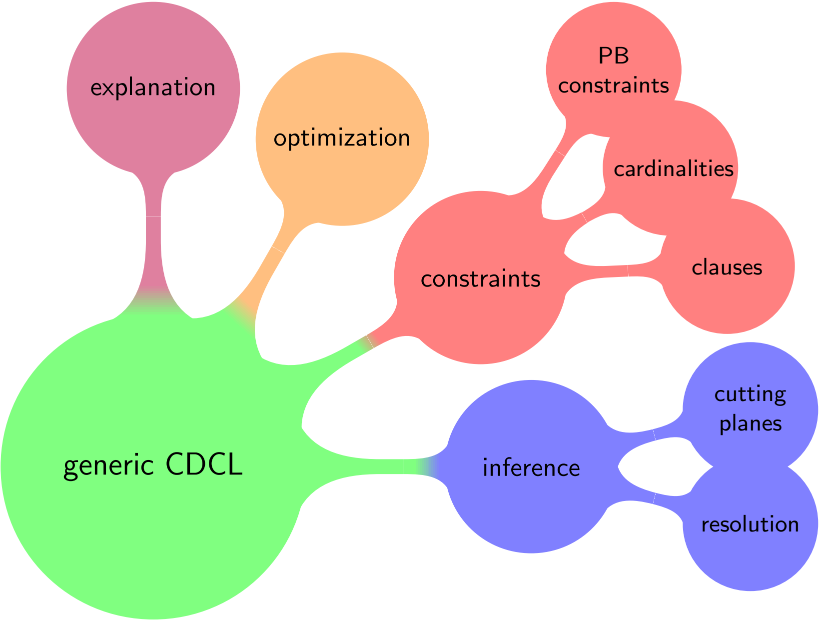
That can solve Optimization problems

An insight of a new solver submitted to PB 2010

Solving more problems : MAXSAT, WBO, etc.

Conclusion

A flexible framework for solving propositional problems



SAT4J today

- SAT4J MAXSAT considered state-of-the-art on Partial [Weighted] MaxSAT application benchmarks (2009).
- SAT4J PB (Res, CP) are not very efficient, but correct (arbitrary precision arithmetic).
- SAT4J SAT solvers can be found in various software from academia (Alloy 4, Forge, ...) to commercial applications (GNA.sim).
- SAT4J PB Res solves Eclipse plugin dependencies since June 2008 (Eclipse 3.4, Ganymede, c.f. LaSh talk July 15)
  - SAT4J ships with every product based on the Eclipse platform (more than 25 millions downloads from Eclipse.org since June 2008)
  - SAT4J helps to build Eclipse products daily (e.g. nightly builds on Eclipse.org, IBM, Oracle, SAP, etc)
  - SAT4J helps to update Eclipse products worldwide daily

Thanks for your attention

http://www.sat4j.org/

Questions ?

UNIVERSITÉ D'ARTOIS