On reusing or adapting SAT solvers for boolean optimization

Daniel Le Berre and Emmanuel Lonca

CRIL CNRS UMR 8188 – Université d'Artois

PoS'13 – 9 July 2013

Presentation plan

SAT solvers

On-the-fly constraints addition

Experiments

Conclusion, perspectives

Presentation plan

SAT solvers

On-the-fly constraints addition

Experiments

Conclusion, perspectives

SAT, from decision problem to iterative processes

- SAT is initially a decision problem
- possible to use a SAT solver for more complex problems than decision ones :
  - Pseudo-Boolean Optimization : bound constraints
  - MaxSat : selector variables + bound constraints
  - CEGAR, model enumeration : blocking clauses
- incremental SAT : keep informations between solver calls
  - successive decision problems : solver stops after each call
    - constraining problem before each new call
    - using assumptions
  - keep learnt clauses

Example : PB optimization

constraints :

$$\begin{aligned} 5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 &\geq 8 \\ 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} &\geq 5 \\ x_1 + x_3 + x_4 &\geq 2 \end{aligned}$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

UNIVERSITÉ D'ARTOIS

Example : PB optimization

constraints :

$$\begin{aligned} 5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 &\geq 8 \\ 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} &\geq 5 \\ x_1 + x_3 + x_4 &\geq 2 \end{aligned}$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5 \quad f(\omega_1) = 5$

UNIVERSITÉ D'ARTOIS

Example : PB optimization

constraints :

$$\begin{aligned} 5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 &\geq 8 \\ 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} &\geq 5 \\ x_1 + x_3 + x_4 &\geq 2 \\ 4x_2 + 2x_3 + x_5 &< 5 \end{aligned}$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5 \quad f(\omega_1) = 5$

Example : PB optimization

constraints :

$$
\begin{array}{rcl}
5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 & \geq & 8 \\
5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} & \geq & 5 \\
x_1 + x_3 + x_4 & \geq & 2 \\
4x_2 + 2x_3 + x_5 & < & 5
\end{array}
$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$$
\begin{array}{lll}
\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5 & f(\omega_1) = & 5 \\
\omega_2 = x_1, \overline{x_2}, x_3, \overline{x_4}, x_5 & f(\omega_2) = & 3
\end{array}
$$

Example : PB optimization

constraints :

$$\begin{aligned} 5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 &\geq 8 \\ 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} &\geq 5 \\ x_1 + x_3 + x_4 &\geq 2 \\ 4x_2 + 2x_3 + x_5 &< 3 \end{aligned}$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$$\begin{aligned} \sout{\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5} \quad \sout{f(\omega_1) = 5} \\ \omega_2 = x_1, \overline{x_2}, x_3, \overline{x_4}, x_5 \quad f(\omega_2) = 3 \end{aligned}$$

Example : PB optimization

constraints :

$$
\begin{array}{rcl}
5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 & \geq & 8 \\
5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} & \geq & 5 \\
x_1 + x_3 + x_4 & \geq & 2 \\
4x_2 + 2x_3 + x_5 & < & 3
\end{array}
$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$$
\begin{array}{lll}
\sout{\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5} & \sout{f(\omega_1) =} & \sout{5} \\
\omega_2 = x_1, \overline{x_2}, x_3, \overline{x_4}, x_5 & f(\omega_2) = & 3 \\
\omega_3 = x_1, \overline{x_2}, \overline{x_3}, x_4, x_5 & f(\omega_3) = & 1
\end{array}
$$

UNIVERSITÉ D'ARTOIS

Example : PB optimization

constraints :

$$\begin{aligned} 5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 &\geq 8 \\ 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} &\geq 5 \\ x_1 + x_3 + x_4 &\geq 2 \\ 4x_2 + 2x_3 + x_5 &< 1 \end{aligned}$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$$\begin{aligned} \sout{\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5} \quad \sout{f(\omega_1) = 5} \\ \sout{\omega_2 = x_1, \overline{x_2}, x_3, \overline{x_4}, x_5} \quad \sout{f(\omega_2) = 3} \\ \omega_3 = x_1, \overline{x_2}, \overline{x_3}, x_4, x_5 \quad f(\omega_3) = 1 \end{aligned}$$

Example : PB optimization

constraints :

$$
\begin{array}{rcl}
5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 & \geq & 8 \\
5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} & \geq & 5 \\
x_1 + x_3 + x_4 & \geq & 2 \\
4x_2 + 2x_3 + x_5 & < & 1
\end{array}
$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

~~$\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5 \quad f(\omega_1) = 5$~~

~~$\omega_2 = x_1, \overline{x_2}, x_3, \overline{x_4}, x_5 \quad f(\omega_2) = 3$~~

$\omega_3 = x_1, \overline{x_2}, \overline{x_3}, x_4, x_5 \quad f(\omega_3) = 1$

$\varnothing$

Example : PB optimization

constraints :

$$\begin{array}{rcl} 5x_1 + 3x_2 + 2x_3 + 2x_4 + x5 & \geq & 8 \\ 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x5} & \geq & 5 \\ x_1 + x_3 + x_4 & \geq & 2 \\ 4x_2 + 2x_3 + x_5 & < & 1 \end{array}$$

objective function : $\min f(\omega) = 4x_2 + 2x_3 + x_5$

models :

$\sout{\omega_1 = \overline{x_1}, x_2, \overline{x_3}, x_4, x_5 \quad f(\omega_1) = 5}$

$\sout{\omega_2 = x_1, \overline{x_2}, x_3, \overline{x_4}, x_5 \quad f(\omega_2) = 3}$

$\omega_3 = x_1, \overline{x_2}, \overline{x_3}, x_4, x_5 \quad f(\omega_3) = 1$

$\varnothing$

ω_3 is an optimal solution

Another way to optimize : on-the-fly constraint addition

- do not stop the solver when a model is found
- old idea : branch & bound (BSOLO)
  $\rightarrow$ implementation in modern CDCL solvers
- generate a falsified bound constraint to take benefit from conflict analysis procedure
- goal : discard stop/restart solver cost

Sat4j and the 12 queens

- solution enumeration with Sat4j :
  1. search for a model $x_1, x_2, ..., x_n$, quit if none
  2. add the constraint $\overline{x_1} \vee \overline{x_2} \vee ... \vee \overline{x_n}$
  3. goto 1

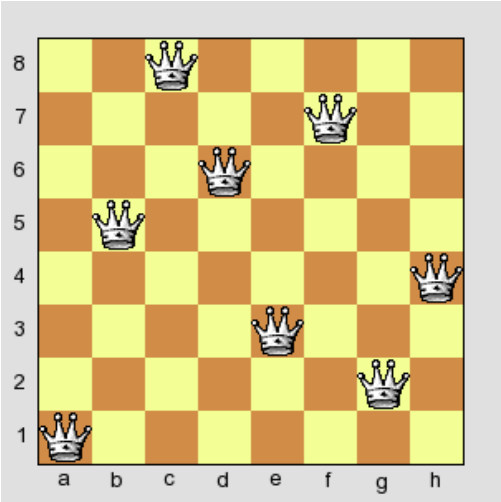


image source : `http://paulbutler.org/archives/n-queens-in-a-tweet/`

- 14200 solutions computation time (black box) : 593s (Abscon : 5s)

UNIVERSITÉ D'ARTOIS

Sat4j and the 12 queens

- solution enumeration with Sat4j :
  1. search for a model $x_1, x_2, ..., x_n$, quit if none
  2. add the constraint $\overline{x_1} \vee \overline{x_2} \vee ... \vee \overline{x_n}$
  3. goto 1

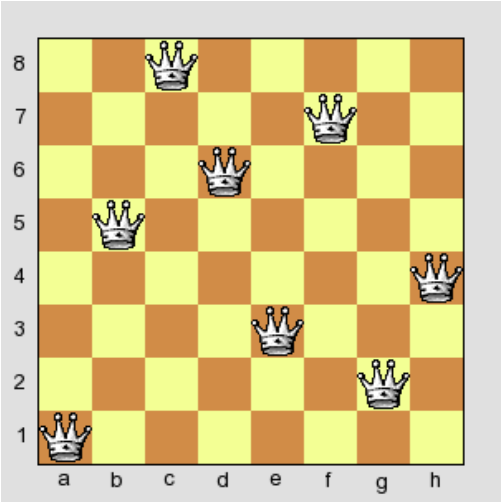


image source : `http://paulbutler.org/archives/n-queens-in-a-tweet/`

- 14200 solutions computation time (black box) : 593s (Abscon : 5s)
- when adding clauses on-the-fly : 113s $\rightarrow$ on-the-fly clauses based enumeration seems efficient

UNIVERSITÉ D'ARTOIS

Sat4j and the 12 queens

- solution enumeration with Sat4j :
  1. search for a model $x_1, x_2, ..., x_n$, quit if none
  2. add the constraint $\overline{x_1} \vee \overline{x_2} \vee ... \vee \overline{x_n}$
  3. goto 1

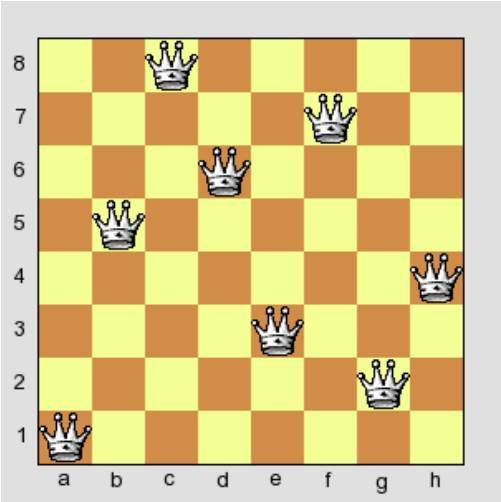


image source : `http://paulbutler.org/archives/n-queens-in-a-tweet/`

- 14200 solutions computation time (black box) : 593s (Abscon : 5s)
- when adding clauses on-the-fly : 113s $\rightarrow$ on-the-fly clauses based enumeration seems efficient

what about optimization ?

Presentation plan

SAT solvers

On-the-fly constraints addition

Experiments

Conclusion, perspectives

UNIVERSITÉ D'ARTOIS

CDCL solvers

- CDCL solver : learn redundant constraints to guide the search
- when a conflict occurs, learn a clause and backtrack
- backtrack goal : put the solver in a state which is compatible with the new constraint (as if this constraint would have been in the initial set of constraints)
- our addition : adding falsified constraints "on-the-fly" to force the solver to backtrack

On-the-fly falsified clause addition

- a clause propagates a unique literal at a unique decision level
- backtrack to the decision level where the constraint was falsified for the first time, and call the conflict analysis procedure
- conflict analysis procedure is sufficient

On-the-fly cardinality constraint addition

- constraints which can be written as $\sum_{i=1}^{n} x_i \leq k$
- a cardinality constraint may propagate multiple literals, but at a unique decision level
- if k literals in n are set to true, others must be set to false
- a specific propagation process is required : the constraint must be able to propagate itself some literals (conflict analysis is not sufficient)

Cardinality constraint : conflict analysis is not sufficient

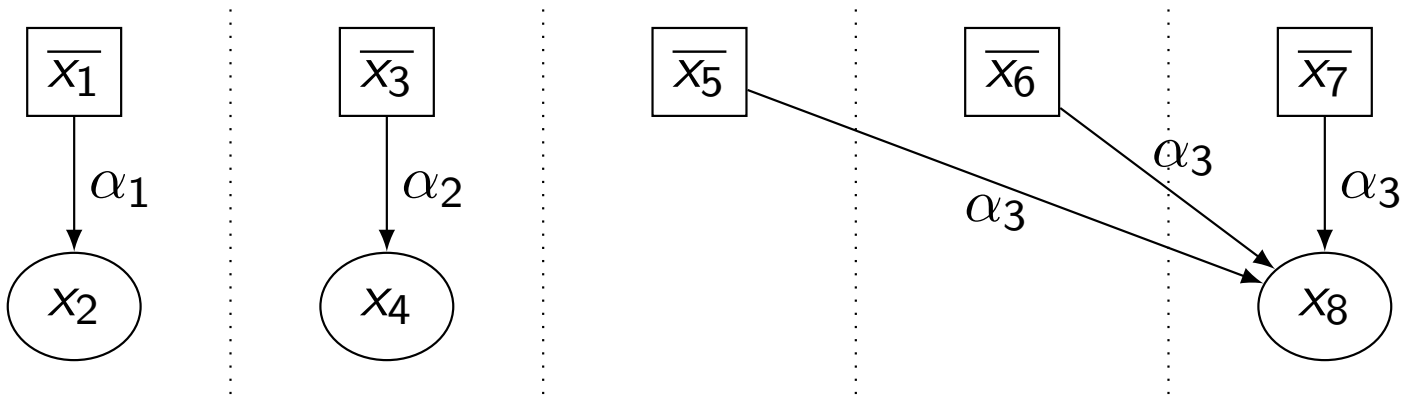
- constraints :

α_1 : $x_1 \vee x_2$

α_2 : $x_3 \vee x_4$

α_3 : $x_5 \vee x_6 \vee x_7 \vee x_8$

- implication graph :



Cardinality constraint : conflict analysis is not sufficient

- constraints :

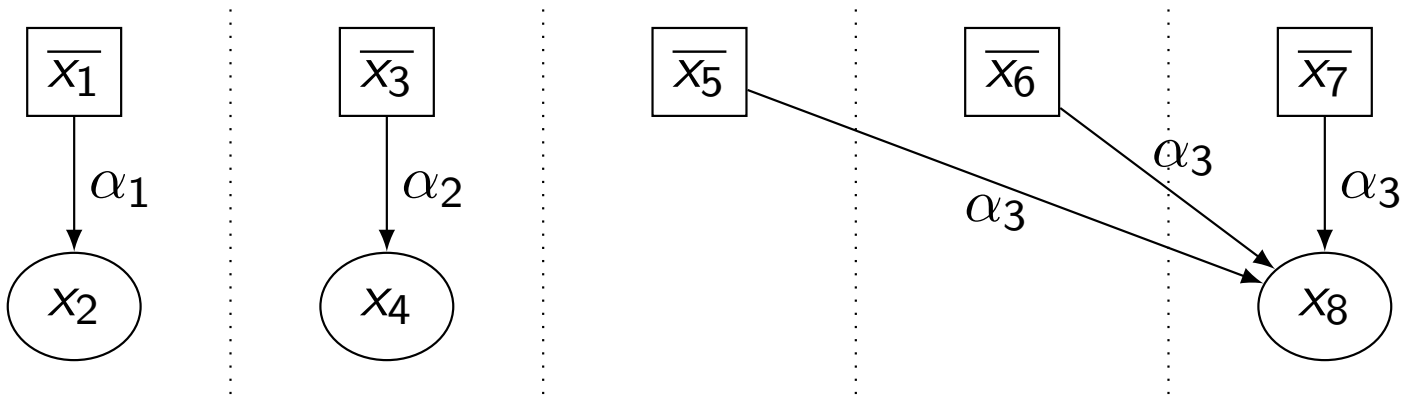
α_1 : $x_1 \vee x_2$

α_2 : $x_3 \vee x_4$

α_3 : $x_5 \vee x_6 \vee x_7 \vee x_8$

β_1 : $x_2 + x_4 + x_7 + x_8 \leq 2$

- implication graph :



- adding falsified constraint $x_2 + x_4 + x_7 + x_8 \leq 2$

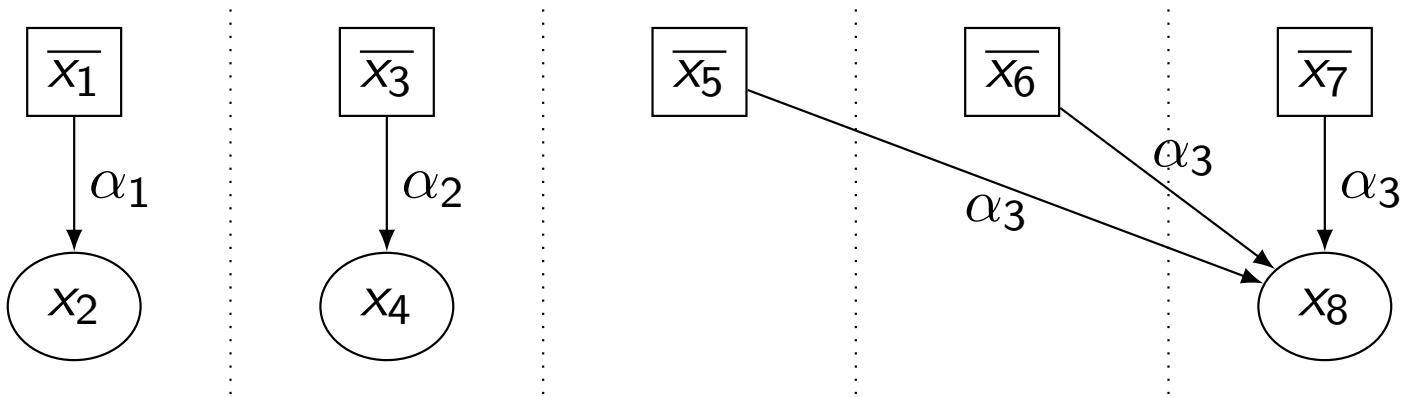
Cardinality constraint : conflict analysis is not sufficient

- constraints :

α_1 : $x_1 \vee x_2$
α_2 : $x_3 \vee x_4$
α_3 : $x_5 \vee x_6 \vee x_7 \vee x_8$

β_1 : $x_2 + x_4 + x_7 + x_8 \leq 2$
γ_1 : $\overline{x_2} \vee \overline{x_4} \vee x_5 \vee x_6 \vee x_7$

- implication graph :



- adding falsified constraint $x_2 + x_4 + x_7 + x_8 \leq 2$
- conflict analysis : $\overline{x_2} \vee \overline{x_4} \vee x_5 \vee x_6 \vee x_7$

Cardinality constraint : conflict analysis is not sufficient

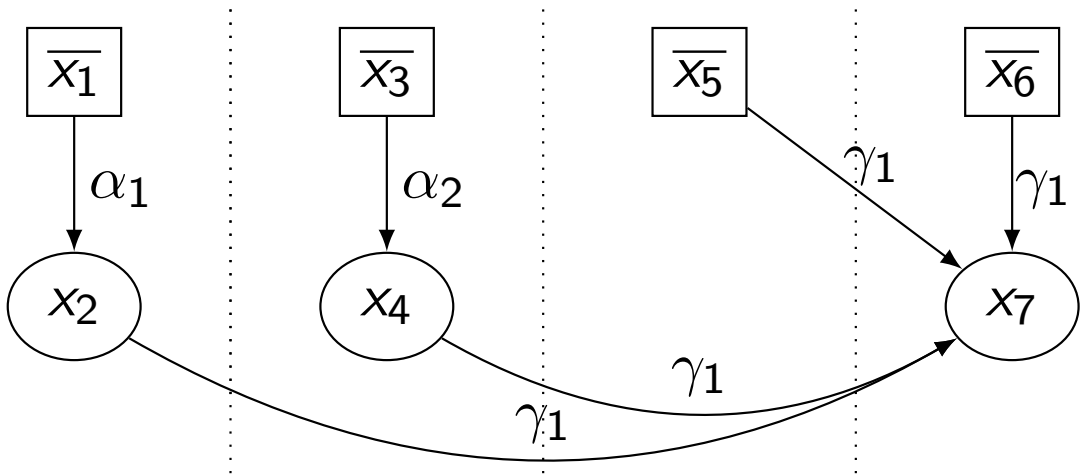
- constraints :

α_1 : $x_1 \vee x_2$ $\quad$ β_1 : $x_2 + x_4 + x_7 + x_8 \leq 2$

α_2 : $x_3 \vee x_4$ $\quad$ γ_1 : $\overline{x_2} \vee \overline{x_4} \vee x_5 \vee x_6 \vee x_7$

α_3 : $x_5 \vee x_6 \vee x_7 \vee x_8$

- implication graph :



- adding falsified constraint $x_2 + x_4 + x_7 + x_8 \leq 2$
- conflict analysis : $\overline{x_2} \vee \overline{x_4} \vee x_5 \vee x_6 \vee x_7$
- β_1 is still falsified !

Cardinality constraint : conflict analysis is not sufficient

- constraints :

α_1 : $x_1 \vee x_2$

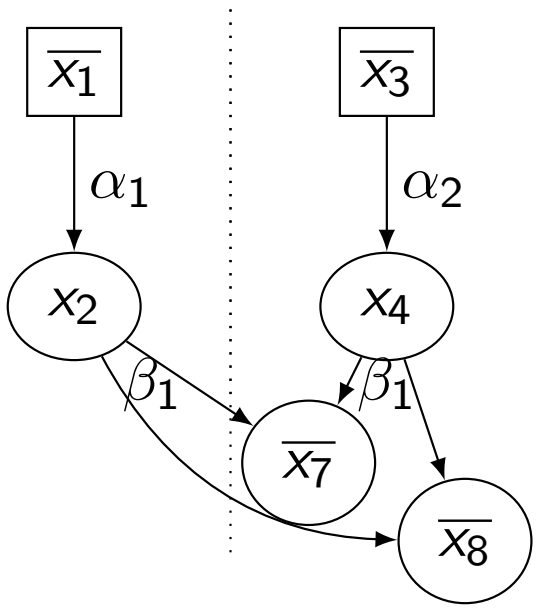
α_2 : $x_3 \vee x_4$

α_3 : $x_5 \vee x_6 \vee x_7 \vee x_8$

β_1 : $x_2 + x_4 + x_7 + x_8 \leq 2$

γ_1 : $\overline{x_2} \vee \overline{x_4} \vee x_5 \vee x_6 \vee x_7$

- implication graph :



- backjump to β_1 assertion level, and propagate $\overline{x_7}$ and $\overline{x_8}$

UNIVERSITÉ D'ARTOIS

On-the-fly pseudo-boolean constraint addition

- constraints which can be written as $\sum_{i=1}^{n} w_i x_i \leq k$
- if the weights sum of the literals set to true is equal to p, other literals whose weights are higher than $k - p$ must be propagated to false
- PB constraints may propagate literals at multiple decision levels
  - $5\overline{x_1} + 2\overline{x_2} + \overline{x_3} \leq 2$
  - propagate x_1 at decision level 0
  - propagate x_2 (resp. x_3) at decision level where $\overline{x_3}$ (resp. $\overline{x_2}$) is propagated
- idea : backtrack to the decision level where the constraint has been falsified for the first time, and analyse the conflict
- a specific propagation process is also required

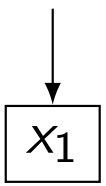
PB constraints : some propagations may be lost

- a model :

PB constraints : some propagations may be lost

- a model :

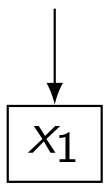
- adding the constraint $5x_1 + 2x_2 + x_3 \geq 6$, black box :



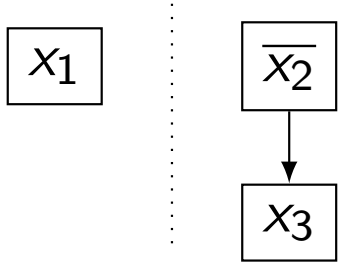
PB constraints : some propagations may be lost

- a model :

- adding the constraint $5x_1 + 2x_2 + x_3 \geq 6$, black box :



- adding the constraint $5x_1 + 2x_2 + x_3 \geq 6$, on the fly :



extended CDCL algorithm for PB-OPT

1. propagate a literal
2. if a model is found, generate a bound constraint, and backjump to its falsification level
3. if a conflict appeared at decision level 0, return OPT
4. if a conflict appeared at another decision level, process conflict analysis, learn the resulting constraint, and apply the specific propagation process if needed
5. goto 1

Presentation plan

SAT solvers

On-the-fly constraints addition

Experiments

Conclusion, perspectives

Experimental protocol

- model enumeration :
  - Intel Quad-core XEON X5550 2,66 GHz, 32 Go de RAM
  - benchmarks used by Morgado et Marques Silva in *Good learning and implicit model enumeration* (ICTAI'05)
  - timeout : 20 minutes
- optimization :
  - Intel XEON 3,0 GHz, 2 Go de RAM
  - benchmarks and timeouts from PB'10 et MAXSAT'10
- both approaches were submitted to MAXSAT'13 evaluation

Model enumeration results

- solved instances (all models enumerated, timeout : 20 minutes) :

cat.	#inst.	black box	on the fly
CBS	5000	4076	**4881**
uf	3700	3110	**3408**
flat	1700	1151	**1675**

- average solving time by category, median time between parenthesis :

cat.	black box	on the fly	ratio
CBS	53239 (1045)	**1584** (**694**)	33.6
uf	18264 (**101**)	**667** (121)	27.4
flat	39730 (2447)	**1672** (**1013**)	23.8

PB'10 results

solved instances (30 minutes ; UNSAT between parenthesis) :

cat.	#inst.	black box	on the fly
BIGINT-LIN	532	**125** (57)	115 (57)
SMALLINT-LIN	699	**270** (33)	266 (33)
SMALLINT-NLC	409	273 (0)	**275** (0)

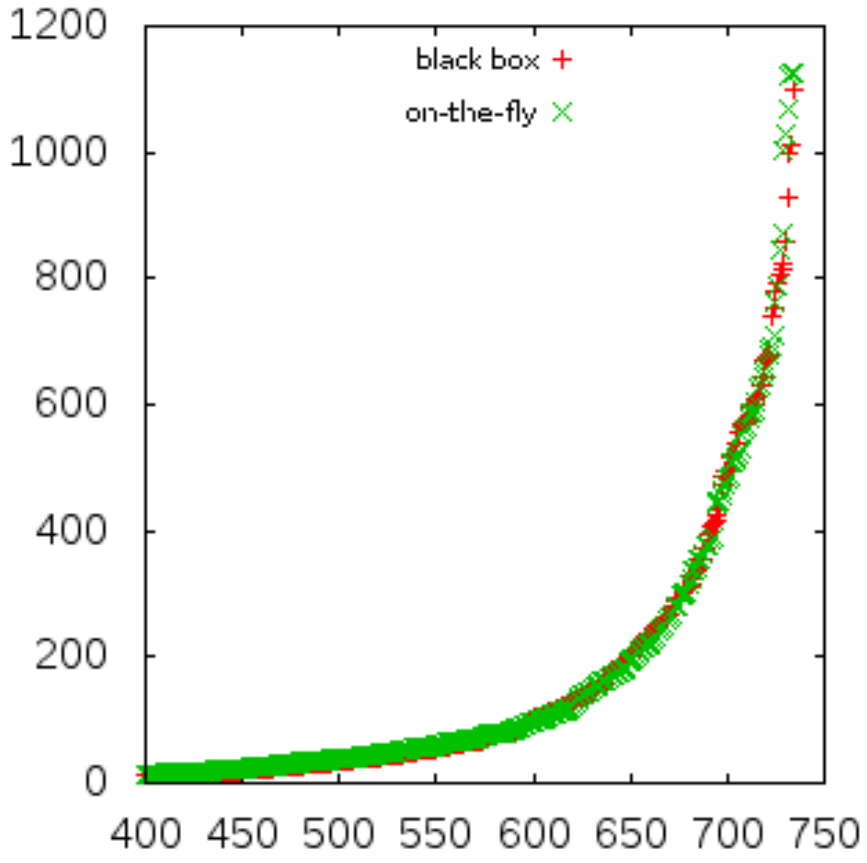
MAXSAT'10 results

solved instances (timeout : 20 minutes) :

cat.	#inst.	black box	on the fly
ms_crafted	167	**2**	**2**
ms_industrial	77	**8**	5
ms_random	300	**0**	**0**
pms_crafted	385	**190**	187
pms_industrial	497	**270**	258
pms_random	240	**26**	**26**
wms_crafted	149	**43**	**43**
wms_random	200	**16**	**16**
wpms_crafted	378	**146**	142
wpms_industrial	132	**36**	35
wpms_random	150	**29**	**29**

MAXSAT'10 cactus plot

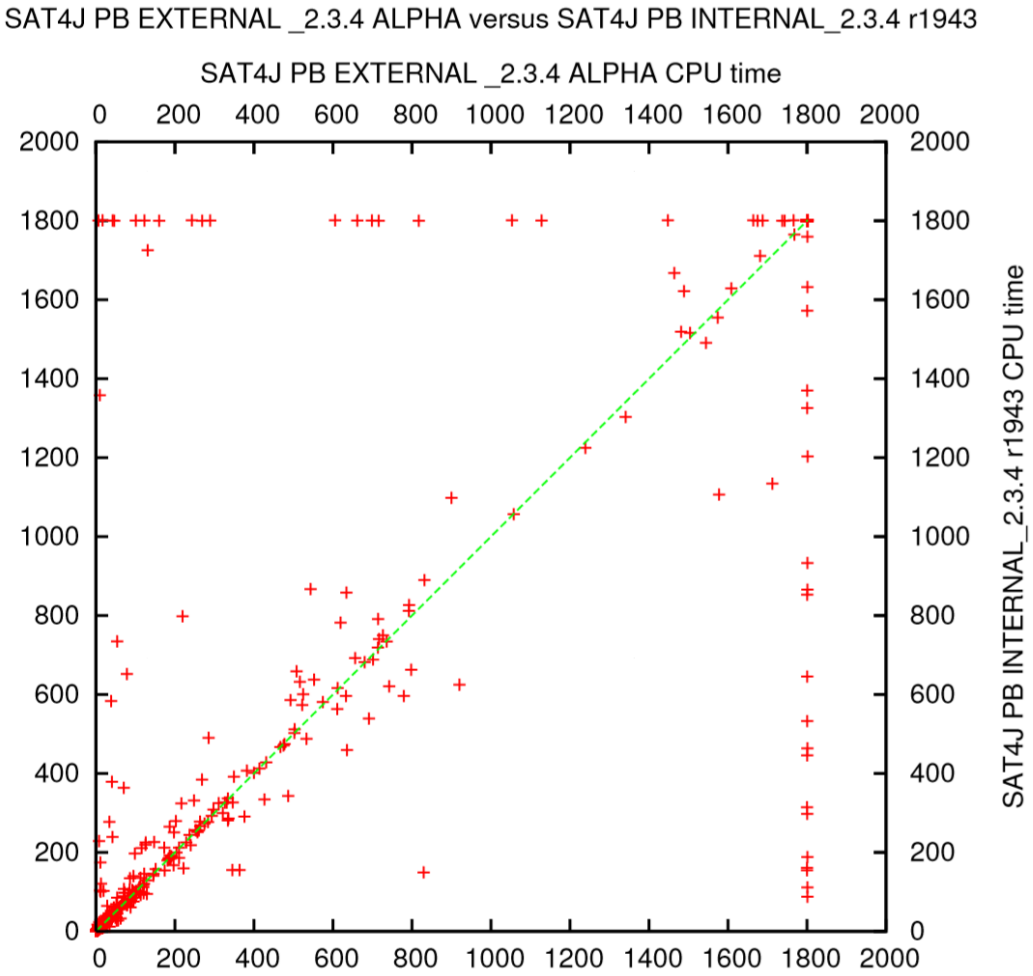
solving time :



UNIVERSITÉ D'ARTOIS

Solving times comparison

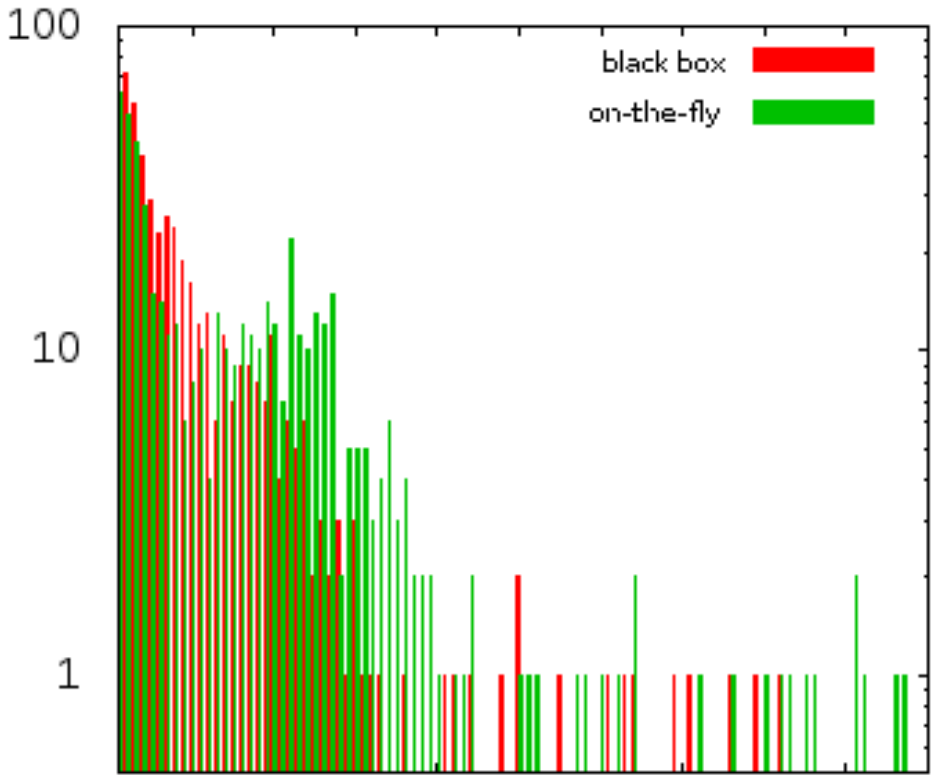
solving times :



UNIVERSITÉ D'ARTOIS

intermediate model count MAXSAT'10

Intermediate model found while searching an optimum one
(x-axis : intermediate solution count, y-axis : instance count) :



UNIVERSITÉ D'ARTOIS

crazy model enumeration example

- sometimes, on-the-fly finds a lot of intermediate solutions
- normalized-rand0c02bc.cudf.paranoid.opb :

black box output :

```
s OPTIMUM FOUND
c Found 29 solution(s)
v x1 -x2 -x3 x4 -x5 x6 -x7 -x8 -x9 x10 ...
c objective function=8730
c Total wall clock time (in seconds) : 658.531
```

on-the-fly output :

```
c Found solution #196 (1630.88)s
c cleaning 58002 clauses out of 116019 with flag
6191114/1760000
c cleaning 58999 clauses out of 118017 with flag
6373293/1820000
o 8748
c Found solution #197 (1760.21)s
c cleaning 60009 clauses out of 120018 with flag
6561648/1881000
o 8747
```

Presentation plan

SAT solvers

On-the-fly constraints addition

Experiments

Conclusion, perspectives

Conclusions, perspectives

- mitigate results :
  - on-the-fly approach seems very effective for model enumeration
  - no gain for optimization :-(
- useful in other cases ? (CEGAR, MUS/MSS enumeration)
- results may be altered by PB constraint handling issue : better handling ?
  - restart after adding some classes of constraints (mixing black box and on-the-fly approaches)
  - learning unit clauses to force restarting for level 0 propagation PB constraints

Thank you !

Thank you for paying attention :-)

UNIVERSITÉ D'ARTOIS