Survey on integrating cutting planes in CDCL solvers

Daniel Le Berre

CNRS, Université d'Artois, FRANCE
{leberre}@cril.univ-artois.fr

Banff, 22 January 2014

Motivation

- CDCL proof system is resolution [PD11, AFT11]
- Resolution in CDCL is used during conflict analysis
- Can we change the conflict analysis procedure to use another proof system ?
- Let's try "cutting planes" to generate linear pseudo boolean constraints

Outline of the talk

Definitions and properties

Handling Pseudo-Boolean constraints instead of clauses

Conflict Driven “cutting planes” reasoning

PB Solvers in practice (PB evaluations)

Outline

Linear Pseudo-Boolean constraints (LPB)

$$\sum_{i=1}^{n} a_i x_i \otimes k$$

- boolean variables x_i are integers taking their value in $\{0, 1\}$ ($x_i \geq 0$ and $x_i \leq 1$)
- $\overline{x_i} = 1 - x$
- coefficients a_i and degree k are integer-valued constants
- $\otimes \in \{<, \leq, =, \geq, >\}$ with ($< k \leftrightarrow \leq k - 1$ and $= k \leftrightarrow \leq k \wedge \geq k$)

Pseudo-Boolean decision problem : satisfying a set of LPB is NP-complete

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

LPB = Concise boolean function representation

- clauses are specific LPB :

$$\bigvee_{i=1}^{n} l_i \quad \equiv \quad \sum_{i=1}^{n} l_i \geq 1 \quad \equiv \quad \sum_{i=1}^{n} \overline{l_i} \leq n-1$$

  $x_1 \vee x_2 \vee x_3$ translates into $x_1 + x_2 + x_3 \geq 1$
  or $\overline{x_1} + \overline{x_2} + \overline{x_3} \leq 2$

- cardinality constraints at least/at most 2 out of $\{x_1, x_2, x_3\}$ translate into

$$x_1 + x_2 + x_3 \geq 2$$

$$x_1 + x_2 + x_3 \leq 2$$

- Knapsack constraint : $\sum w_i.x_i \leq W$
- Subset sum constraint : $\sum a_i.x_i = k$

Linear Pseudo Boolean constraints normalization

Representation used when designing a solver

- remember that $x = 1 - \overline{x}$
- usual form : $\geq$ inequality and positive constants

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

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

$$\equiv 3x_1 + -4(1 - \overline{x_2}) + 7x_3 + -(1 - \overline{x_4}) \geq 5$$

$$\equiv 3x_1 + 4\overline{x_2} + 7x_3 + \overline{x_4} \geq 10$$

- note that

$$x_1 + x_2 + x_3 + x_4 + x_5 \leq 1$$

is represented

$$\overline{x_1} + \overline{x_2} + \overline{x_3} + \overline{x_4} + \overline{x_5} \geq 4$$

Basic operations on Linear inequalities

addition:
$$\frac{\sum_i a_i.x_i \geq k \quad \sum_i a'_i.x_i \geq k'}{\sum_i (a_i + a'_i).x_i \geq k + k'}$$

linear combination:
$$\frac{\sum_i a_i.x_i \geq k \quad \sum_i a'_i.x_i \geq k'}{\sum_i (\alpha.a_i + \alpha'.a'_i).x_i \geq \alpha.k + \alpha'.k' \quad \text{with } \alpha > 0 \text{ and } \alpha' > 0}$$

division:
$$\frac{\sum_i a_i.x_i \geq k \quad \alpha > 0}{\sum_i \frac{a_i.x_i}{\alpha} \geq \frac{k}{\alpha}}$$

TCS division

$$\textit{TCS division:}\quad \frac{\sum_i \alpha.a_i.x_i \geq k \quad \alpha > 0}{\sum_i a_i.x_i \geq \lceil \frac{k}{\alpha} \rceil}$$

$$\textit{tcs division:}\quad \frac{2x_2 + 2x_3 + 2x_4 \geq 3}{\begin{array}{c} x_2 + x_3 + x_4 \geq \lceil 3/2 \rceil \\ x_2 + x_3 + x_4 \geq 2 \end{array}}$$

ILP division (Chvátal-Gomory cut)

- When the variables x_i and degree k are integer
- Removes some non integral part of the cut

$$\textit{ILP division:} \quad \frac{\sum_i a_i.x_i \geq k \quad \alpha > 0}{\sum_i \lceil \frac{a_i}{\alpha} \rceil .x_i \geq \lceil \frac{k}{\alpha} \rceil}$$

$$\frac{5x_3 + 3x_4 \geq 5}{\lceil 5/5 \rceil x_3 + \lceil 3/5 \rceil x_4 \geq \lceil 5/5 \rceil}$$
$$x_3 + x_4 \geq 1$$

One can always reduce a LPB constraint to a clause !

Clashing linear combination

Also called Gaussian or Fourier–Motzkin elimination

- Apply linear combination between LPB constraints with at least one opposite literal.
- Generalization of resolution [Hoo88]

clashing combination:

$$\frac{\begin{array}{c}\sum_i a_i.x_i + \alpha' \sum_{j=1}^{m} y_j \geq k \\ \sum_i a'_i.x_i + \alpha \sum_{j=1}^{m} \overline{y_j} \geq k'\end{array}}{\begin{array}{c}\sum_i (\alpha.a_i + \alpha'.a'_i).x_i \geq \alpha.k + \alpha'.k' - \alpha.\alpha'.m \\ \text{with } \alpha > 0 \text{ and } \alpha' > 0\end{array}}$$

$$\frac{x_1 + x_2 + 3x_3 + x_4 \geq 3 \qquad 2\overline{x_1} + 2\overline{x_2} + x_4 \geq 3}{\begin{array}{c}2x_1 + 2x_2 + 6x_3 + 2x_4 + 2\overline{x_1} + 2\overline{x_2} + x_4 \geq 2 \times 3 + 3 \\ 2x_1 + 2x_2 + 6x_3 + 2x_4 + 2 - 2x_1 + 2 - 2x_2 + x_4 \geq 9 \\ 6x_3 + 3x_4 \geq 5\end{array}}$$

Note that $2x + 2\overline{x} = 2$, not 0 !

Note that the coefficients are growing !

Some remarks about clashing combination

- Clashing combination looks like resolution ?

$$\frac{x_1 + x_3 + x_4 \geq 1 \qquad \overline{x_1} + x_2 + x_5 \geq 1}{x_2 + x_3 + x_4 + x_5 \geq 1}$$

- What about common literals ?

$$\frac{x_1 + x_2 + x_3 + x_4 \geq 1 \qquad \overline{x_1} + x_2 + x_4 \geq 1}{2x_2 + x_3 + 2x_4 \geq 1}$$

- With more than one variable ?

$$\frac{x_1 + x_2 + x_3 + x_4 \geq 1 \qquad \overline{x_1} + \overline{x_2} + x_4 \geq 1}{x_3 + 2x_4 \geq 0}$$

Saturation

coefficients can be trimmed to the value of the degree

$$\textit{saturation:}\quad \frac{\sum_i a_i.x_i + \sum_j b_j.y_j \geq k \qquad b_j > k}{\sum_i a_i.x_i + \sum_j k.y_j \geq k}$$

$$\frac{6x_3 + 3x_4 \geq 5}{5x_3 + 3x_4 \geq 5}$$

$$\frac{2x_2 + x_3 + 2x_4 \geq 1}{x_2 + x_3 + x_4 \geq 1}$$

[iterated application of ILP division rule with $\frac{k+1}{k} \leq \alpha < \frac{k}{k-1}$]

Weakening

We can reduce the degree of the constraint by "satisfying" any of its literals

$$\textit{weakening:} \quad \frac{\sum_{i \neq j} a_i.x_i + a_j.x_j \geq k}{\sum_{i \neq j} a_i.x_i \geq k - a_j}$$

$$\frac{5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8}{3x_2 + 2x_3 + 2x_4 + x_5 \geq 3}$$

Useful for reducing the value of the degree !

[Apply linear combination rule with $\overline{x_j} \geq 0$]

Reduction to cardinality

Extract a cardinality constraint from a LPB constraint

$$\textit{reduce to card:} \quad \frac{\begin{array}{c}\sum_{i=1}^{n} a_i.x_i \geq k \\ a_1 \geq a_2 \geq \ldots a_n\end{array}}{\begin{array}{c}\sum_{i=1}^{n} x_i \geq k' \\ \text{with } \sum_{i=1}^{k'-1} a_i < k \leq \sum_{i=1}^{k'} a_i\end{array}}$$

$$\frac{5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8}{x_1 + x_2 + x_3 + x_4 + x_5 \geq 2}$$

The various Cutting Planes

- Linear combination + ILP division = Chvátal-Gomory ILP cutting planes
- Addition + TCS division = Proof complexity cutting planes
- Linear clashing combination + saturation = Hooker's generalized resolution cutting planes

Integrating Cutting Planes in a CDCL solver : replace Resolution during Conflict Analysis by Hooker's Cutting Planes

Outline

Requirements for constraints in a CDCL solver

- Detect falsified state
- Detect propagation of literals
- Provide a "reason" during conflict analysis

Some remarks about clauses

$$l_1 \vee l_2 \vee ... \vee l_n$$

- Falsified when all its literals are falsified

$$l_1 \vee l_2 \vee ... \vee l_n$$

- Propagates when all but one literals are falsified

$$l_1 \vee l_2 \vee ... \vee l_n$$

- Propagates one literal
- Appears at most once as a reason for an assignment

Chaff : 2 watched literals per clause

Some remarks about cardinality constraints

$$l_1 + l_2 + ... + l_n \geq k$$

- Falsified when at least $n - k + 1$ literals are falsified

$$l_1 + l_2 + l_3 + l_4 + l_5 + l_6 \geq 4$$

Note unassigned literals !

- Propagates when exactly $n - k$ literals are falsified

$$l_1 + l_2 + l_3 + l_4 + l_5 + l_6 \geq 4$$

- Propagates k literals
- Appears at most once as a reason for at most k consecutive assignments.

Extended $k + 1$ watched literals per cardinality

Some remarks about LBP constraints

$$a_1.l_1 + a_2.l_2 + ... + a_n.l_n \geq k$$

$$A = \sum_i a_i$$

Slack s : $A - k - \sum_{l_i falsified} a_i$

- Falsified when $s < 0$ (depends on falsified literals)

$$5l_1 + 3l_2 + 2l_3 + l_4 + l_5 + l_6 \geq 6$$

- Propagates remaining literals when $s = 0$ (*)

$$5l_1 + 3l_2 + 2l_3 + l_4 + l_5 + l_6 \geq 6$$

- Propagates literals x_i for which $s < a_i$
- May appear several times as a reason for non consecutive assignments (*)

Extended watched literals based on coefficients !

Watched Literals for LPB constraints

Described in Galena [CK03] and BChaff [Par04], may have already existed in PBS or Satzoo.

- General case :
  Let $M = max(a_i)$
  NbWatch = minimal number of literals x_i such that
  $\sum a_i \geq k + M$.
- Cardinality constraints :
  $M = 1$
  $NbWatch = k + 1$
- Clauses :
  $M = 1$
  $k = 1$
  $NbWatch = 2$

Watched literals : consequences

- In LPB constraints, the number of WL is varying during the search.
- In cardinality constraints, the greater the degree, the greater the number of WL.
- Clauses are the best case !
- Big difference for LPB constraint learning

Forced truth values : Implicative and Assertive constraints

- *unit clause* : a clause that propagates one truth value to be satisfiable
- *implicative constraint* : a constraint which propagates *at least* one truth value to be satisfiable.
- a LPB constraint C is *implicative* iff $\exists a_i x_i \in C$ such that $\sum_{j \neq i} a_j < k$ or $\sum a_j - k < a_i$.

Forced truth values : Implicative and Assertive constraints

- *unit clause* : a clause that propagates one truth value to be satisfiable
- *implicative constraint* : a constraint which propagates *at least* one truth value to be satisfiable.
- a LPB constraint C is *implicative* iff $\exists a_i x_i \in C$ such that $\sum_{j \neq i} a_j < k$ or $\sum a_j - k < a_i$.

Example

$$4x_1 + 3x_2 + x_3 + x_4 \geq 8$$

propagates x_1 and x_2

- $3 + 1 + 1 < 8$ so x_1 must be satisfied, same thing on $3x_2 + x_3 + x_4 \geq 4$.

Forced truth values : Implicative and Assertive constraints

- *unit clause* : a clause that propagates one truth value to be satisfiable
- *implicative constraint* : a constraint which propagates *at least* one truth value to be satisfiable.
- a LPB constraint C is *implicative* iff $\exists a_i x_i \in C$ such that $\sum_{j \neq i} a_j < k$ or $\sum a_j - k < a_i$.

Example

$$4x_1 + 3x_2 + x_3 + x_4 \geq 8$$

propagates x_1 and x_2

- $3 + 1 + 1 < 8$ so x_1 must be satisfied, same thing on $3x_2 + x_3 + x_4 \geq 4$.
- One can note that $\sum a_j - k = 1$ so any literal x_i with a coef greater than 1 must be propagated.

Forced truth values : Implicative and Assertive constraints

- *unit clause* : a clause that propagates one truth value to be satisfiable
- *implicative constraint* : a constraint which propagates *at least* one truth value to be satisfiable.
- a LPB constraint C is *implicative* iff $\exists a_i x_i \in C$ such that $\sum_{j \neq i} a_j < k$ or $\sum a_j - k < a_i$.

Example

$$4x_1 + 3x_2 + x_3 + x_4 \geq 8$$

propagates x_1 and x_2

- $3 + 1 + 1 < 8$ so x_1 must be satisfied, same thing on $3x_2 + x_3 + x_4 \geq 4$.
- One can note that $\sum a_j - k = 1$ so any literal x_i with a coef greater than 1 must be propagated.
- Rewrite into $x_1 \wedge x_2 \wedge (x_3 + x_4 \geq 1)$?

Outline

Problems with the integration of Cutting Planes

- Derived LPB constraint must be redondant (logical consequence)
  no problem here
- Derived LPB constraint must be falsified at current decision level
  free for resolution, requires special care for CP
- Derived LPB constraint must be assertive at backtrack level
  syntactical test for clauses, not for PB constraints

Computing the backtrack level

- Just a *max* for clauses
- More complicated for LPBC : an LPB constraint may be assertive at different backtrack levels.
  - Decision literals are no longer "UIP" !
  - Need to backtrack to the first one

Example

Given the decisions $x_1, \neg x_2, \neg x_3$
and the falsified LBP $3x_1 + 2x_2 + x_3 + x_4 \geq 5$.
Where should I backtrack ?

Computing the backtrack level

- Just a *max* for clauses
- More complicated for LPBC : an LPB constraint may be assertive at different backtrack levels.
  - Decision literals are no longer "UIP" !
  - Need to backtrack to the first one

Example

Given the decisions $x_1, \neg x_2, \neg x_3$
and the falsified LBP $3x_1 + 2x_2 + x_3 + x_4 \geq 5$.
Where should I backtrack ?
backtrack to $x_1, \neg x_2$ to propagate x_3 and x_4 ?

Computing the backtrack level

- Just a *max* for clauses
- More complicated for LPBC : an LPB constraint may be assertive at different backtrack levels.
  - Decision literals are no longer "UIP" !
  - Need to backtrack to the first one

Example

Given the decisions $x_1, \neg x_2, \neg x_3$
and the falsified LBP $3x_1 + 2x_2 + x_3 + x_4 \geq 5$.
Where should I backtrack ?
backtrack to $x_1, \neg x_2$ to propagate x_3 and x_4 ?
or to decision level 0 to propagate x_1 ?

Computing an assertive clause

- Let C be a falsified constraint
- $S = lit(C)_{>dl}$
- $D = lit(C)_{=dl}$

1 Pick the reason R for the latest assignment a in C

2 Compute $S = S \cup lit(R)_{>dl}$ and $D = D \cup lit(R)_{=dl} \setminus \{a\}$

- Repeat $1 - 2$ until $|D| = 1$

Computing an assertive LPB constraint

1. Let C be a falsified constraint
2. Pick the reason R for the latest assignment a in C
3. compute α and α' to remove a from C.
4. Weaken R if needed to ensure that the LPB constraint generated by applying linear combination is falsified (reduction)
5. Apply clashing combination : $C = CC(C, R, \alpha, \alpha')$
6. Apply saturation
7. Update the slack of the generated constraint
8. Repeat 2-7 until the slack is 0

Use arbitrary precision arithmetic to prevent overflow

Computing an assertive LPB constraint

1. Let C be a falsified constraint
2. Pick the reason R for the latest assignment a in C
3. compute α and α' to remove a from C.
4. Weaken R if needed to ensure that the LPB constraint generated by applying linear combination is falsified (reduction)
5. Apply clashing combination : $C = CC(C, R, \alpha, \alpha')$
6. Apply saturation
7. Update the slack of the generated constraint
8. Repeat 2-7 until the slack is 0

Use arbitrary precision arithmetic to prevent overflow
Not needed if reduced to cardinality constraint

Example

$$\begin{cases} (C_1) & 5x_1 + 3x_2 + 2x_3 + 2x_4 + x_5 \geq 8 \\ (C_2) & 5\overline{x_1} + 3\overline{x_2} + 2\overline{x_3} + 2\overline{x_4} + \overline{x_5} \geq 5 \\ (C_3) & x_1 + x_3 + x_4 \geq 2 \end{cases}$$

$$\neg x_5^0, x_1^0[C_1], \neg x_4^1, x_3^1[C_3], x_2^1[C_1]$$

$Poss(C_1) = +2$, $Poss(C_2) = -2$

Red. x_1 : (C_1') $\quad 3x_2 + 2x_3 + 2x_4 + x_5 \geq 3$ **poss=+2**

Red. x_3 : (C_1'') $\quad x_2 + x_4 + x_5 \geq 1$ **poss=0**

$CC(C_2, 3 \times C_1'') = 2\overline{x_1}^0 + 2\overline{x_3}^1 + x_4^1 + 2x_5^0 \geq 2$

Assertive at decision level 0 (x_3 is propagated to 1).

Would learn $\overline{x_1} + x_4 + x_5 \geq 1$ with clause learning.

Assertive at decision level 0 (x_4 is propagated to 1).

Outline

A brief history of LPB constraints within SAT solvers

[Bar95] DPLL extension to LPB [opbdp]
[Wal97] (and [Pre02, Pre04]) local search for LPB
[MFSO97] B'n'B LPB solver (GRASP) [bsolo]
[WKS01] incremental SAT with LPB (GRASP) [satire]
[ARMS02, Sak03] LPB contraints with Chaff/CDCL solver [pbs, see also satzoo (minisat)]
[Gin02] extended RelSAT to LPB (LPB learning)
[CK03] CDCL with LPB learning [galena]
[Par04] describe a generic CDCL solver based on group theory handling arbitrary boolean gates.
[SS06] CDCL solver able to learn temporary LPB constraints [pueblo]

A brief history of LPB constraints within SAT solvers

[Bar95] DPLL extension to LPB [opbdp]
[Wal97] (and [Pre02, Pre04]) local search for LPB
[MFSO97] B'n'B LPB solver (GRASP) [bsolo]
[WKS01] incremental SAT with LPB (GRASP) [satire]
[ARMS02, Sak03] LPB contraints with Chaff/CDCL solver [pbs, see also satzoo (minisat)]
[Gin02] extended RelSAT to LPB (LPB learning)
[CK03] CDCL with LPB learning [galena]
[Par04] describe a generic CDCL solver based on group theory handling arbitrary boolean gates.
[SS06] CDCL solver able to learn temporary LPB constraints [pueblo]
After 2006 ?

A brief history of LPB constraints within SAT solvers

[Bar95] DPLL extension to LPB [opbdp]
[Wal97] (and [Pre02, Pre04]) local search for LPB
[MFSO97] B'n'B LPB solver (GRASP) [bsolo]
[WKS01] incremental SAT with LPB (GRASP) [satire]
[ARMS02, Sak03] LPB contraints with Chaff/CDCL solver [pbs, see also satzoo (minisat)]
[Gin02] extended RelSAT to LPB (LPB learning)
[CK03] CDCL with LPB learning [galena]
[Par04] describe a generic CDCL solver based on group theory handling arbitrary boolean gates.
[SS06] CDCL solver able to learn temporary LPB constraints [pueblo]
After 2006 ? Major work on CNF encoding of cardinality and LBP constraints (Minisat+ effect)

SAT4J Pseudo

- Implements the LPB learning described in PBChaff [Gin02] and Galena[CK03]
  - Cardinality learning preferred to LPB learning
  - No management of integer overflow
  - Solvers no longer developed
- Two versions available : resolution based inference or Hooker's generalized resolution "cutting planes" based inference.

The Pseudo Boolean evaluations

http ://www.cril.univ-artois.fr/PB12/

- Organized by Olivier Roussel and Vasco Manquinho from 2005 to 2012
- Uniform input format : OPB files
- Independent assessment of the PB solvers
- Detailed results available for each solver
- Various technologies used since 2006

The solvers

Summary of features

	Mini-Sat+	SAT-4J	Pue-blo	PBS	Bsolo	glpPB
Input	Clauses	LPBC	LPBC	LPBC	LPBC	LPBC
Inference	Res.	C.P. (Hoo-ker)	Mixed	Mixed	Mixed	Full C.P. (MILP)
Optimization	L.S.	L.S.	L.S.	L.S.	B'n'B	Sim-plex

Partial results of the PB07 evaluation

	Mini-Sat+	SAT4J C.P.	Pue-blo 1.4	PBS4-v2	Bsolo	SAT4J Res.	
Dec.	170	88	**202**	197	137	167	UNS
(#371)	**146**	56	139	130	141	110	SAT
Opt S	35	**47**	31	23	33	35	UNS
(#807)	208	147	180	141	**237**	156	OPT
	302	397	389	**416**	325	376	SAT
Opt B	38	38	-	-	29	**40**	UNS
(#388)	36	60	-	-	25	**78**	OPT
	67	96	-	-	92	**112**	SAT

See `http://www.cril.univ-artois.fr/PB07/results/` for details

Partial results of the PB06 evaluation

	#	MSat+	SAT4J	Pueblo	PBS	Bsolo	glpPB
SAT/UNSAT							
pigeon	20	2	20	13	20	2	20
queens	100	100	18	99	100	100	100
tsp	100	91	20	100	85	40	42
fpga	57	35	43	57	47	9	26
uclid	50	47	30	42	44	38	10
OPT SMALLINT							
minprime	156	124	104	118	103	106	52
red.-mps	273	46	70	63	27	54	58
OPT BIGINT							
factor.	100	14	52	-	-	7	-
Ardal (subset sum problem)							
Ardal_1	12	10	2	0	3	2	0

ee http://www.cril.univ-artois.fr/PB06/results/ for details

UNIVERSITÉ D'ARTOIS

LPB constraints case : what can go wrong

Boolean propagation lazy data structure for maintaining an alert value require more bookkeeping than for clauses.

Assertive constraints cannot syntactically be identified.

Linear combination between two conflictual constraints doesn't necessary result in a falsified constraint ! Weakening may be needed to obtain a cutting plane.

Coefficient management In some cases, the coefficients of the LPB keep growing.

Consequence : learning PB constraints does slow down the solver !

Solutions :

- Reduce learned clauses to Cardinality constraints (Galena, PBChaff)
- Learn both a clause and a PB constraint, then eventually remove the PB constraint (Pueblo).
- Learn clauses (Minisat+, PBS).

Trace of the search for pigeon-hole 4/3

```
* #variable= 12 #constraint= 7
* beginMapping
*  1=P1H1  2=P1H2  3=P1H3
*  4=P2H1  5=P2H2  6=P2H3
*  7=P3H1  8=P3H2  9=P3H3
* 10=P4H1 11=P4H2 12=P4H3
* endMapping
+1 x1 +1 x2 +1 x3 >= 1 ;
+1 x4 +1 x5 +1 x6 >= 1 ;
+1 x7 +1 x8 +1 x9 >= 1 ;
+1 x10 +1 x11 +1 x12 >= 1 ;
+1 x1 +1 x4 +1 x7 +1 x10 <= 1 ;
+1 x2 +1 x5 +1 x8 +1 x11 <= 1 ;
+1 x3 +1 x6 +1 x9 +1 x12 <= 1 ;
```

Conclusion

- It is possible to introduce some kind of cutting planes reasoning in CDCL solvers
- Cuts derivation is driven by conflict analysis
- Solves PHP instances expressed by cardinalities (not CNF)
- But in practice learning LPB often slows down the solver
- Learning can be reduced to cardinality constraints
- Recent years focussed on encoding those constraints into CNF
- For a complete view of all the techniques available : Handbook of SAT chapter on LPB and cardinality constraints [RM09]


Albert Atserias, Johannes Klaus Fichte, and Marc Thurley.
Clause-learning algorithms with many restarts and bounded-width resolution.
J. Artif. Intell. Res. (JAIR), 40 :353–373, 2011.


F. Aloul, A. Ramani, I. Markov, and K. Sakallah.
Generic ILP versus Specialized 0-1 ILP : an update.
In *Proceedings of ICCAD'02*, pages 450–457, 2002.


Peter Barth.
A Davis-Putnam based enumeration algorithm for linear pseudo-Boolean optimization.
Technical Report MPI–I–95–2–003, Max-Plank-Institut fur Informatik, Saarbrücken, 1995.


Donald Chai and Andreas Kuehlmann.
A fast pseudo-boolean constraint solver.

In *ACM/IEEE Design Automation Conference (DAC'03)*, pages 830–835, Anaheim, CA, 2003.


Heidi E. Dixon Matthew L. Ginsberg.
Inference methods for a pseudo-boolean satisfiability solver.
In *Proceedings of The Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, pages 635–640, 2002.


J. N. Hooker.
Generalized resolution and cutting planes.
Ann. Oper. Res., 12(1-4) :217–239, 1988.


Vasco M. Manquinho, Paulo F. Flores, João P. Marques Silva, and Arlindo L. Oliveira.
Prime implicant computation using satisfiability algorithms.
In *ICTAI*, pages 232–239, 1997.


Heidi E. Dixon Matthew L. Ginsberg Andrew J. Parkes.

Generalizing boolean satisfiability i : Background and survey of existing work.
In *Journal of Artificial Intelligence Research 21*, 2004.


Knot Pipatsrisawat and Adnan Darwiche.
On the power of clause-learning sat solvers as resolution engines.
Artif. Intell., 175(2) :512–525, 2011.


S. Prestwich.
Randomised backtracking for linear pseudo-boolean constraint problems.
In *Proceedings of Fourth International Workshop on Integration of AI and OR techniques in Constraint Programming for Combinatorial Optimisation Problems (CP-AI-OR'2002)*, pages 7–20, 2002.


S. Prestwich.

Incomplete dynamic backtracking for linear pseudo-boolean problems : Hybrid optimization techniques.
Annals of Operations Research, 130(1-4) :57–73, August 2004.


Olivier Roussel and Vasco M. Manquinho.
Pseudo-boolean and cardinality constraints.
In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 695–733. IOS Press, 2009.


Fadi A. Aloul Arathi Ramani Igor L. Markov Karem A. Sakallah.
Symmetry-breaking for pseudo-boolean formulas.
In *International Workshop on Symmetry on Constraint Satisfaction Problems (SymCon)*, pages 1–12, County Cork, Ireland, 2003.


Hossein M. Sheini and Karem A. Sakallah.

Pueblo : A Hybrid Pseudo-Boolean SAT Solver.
Journal on Satisfiability, Boolean Modeling and Computation (JSAT), 2 :165–182, 2006.


J. P. Walser.
Solving Linear Pseudo-Boolean Constraint Problems with Local Search.
In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 269–274, 1997.


Jesse Whittemore, Joonyoung Kim, and Karem A. Sakallah.
Satire : A new incremental satisfiability engine.
In *DAC*, pages 542–545. ACM, 2001.