Tuning Sat4j PB Solvers for Decision Problems

Romain Wallon
Zoom Seminar – August 28th, 2020
CRIL, Univ Artois & CNRS
Pseudo-Boolean (PB) Constraints

PB solvers generalize SAT solvers to take into account:

- normalized PB constraints: \( \sum_{i=1}^{n} a_i l_i \geq d \)
- cardinality constraints: \( \sum_{i=1}^{n} l_i \geq d \)
- clauses: \( \sum_{i=1}^{n} l_i \geq 1 \equiv \bigvee_{i=1}^{n} l_i \)

in which:

- the coefficients \( a_i \) are non-negative integers
- each \( l_i \) is a literal, i.e., a variable \( v \) or its negation \( \bar{v} = 1 - v \)
- the degree \( d \) is a non-negative integer
The generalized resolution proof system [Hooker, 1988] is used as the counterpart of the resolution proof system in PB solvers such as Sat4j.

\[
\frac{al + \sum_{i=1}^{n} a_il_i \geq d_1}{\sum_{i=1}^{n} (ba_i + ab_i)l_i \geq bd_1 + ad_2 - ab} \quad \text{(cancellation)}
\]

\[
\frac{\sum_{i=1}^{n} a_il_i \geq d}{\sum_{i=1}^{n} \min(a_i, d)l_i \geq d} \quad \text{(saturation)}
\]
Generalized Resolution

The generalized resolution proof system [Hooker, 1988] is used as the counterpart of the resolution proof system in PB solvers such as *Sat4j*

\[
\frac{al + \sum_{i=1}^{n} a_i l_i \geq d_1}{\sum_{i=1}^{n} (ba_i + ab_i) l_i \geq bd_1 + ad_2 - ab} \quad \text{(cancellation)}
\]

\[
\frac{\sum_{i=1}^{n} a_i l_i \geq d}{\sum_{i=1}^{n} \min(a_i, d) l_i \geq d} \quad \text{(saturation)}
\]
Generalized Resolution

The generalized resolution proof system [Hooker, 1988] is used as the counterpart of the resolution proof system in PB solvers such as Sat4j.

\[
\begin{align*}
  al + \sum_{i=1}^{n} a_i l_i & \geq d_1 \\
  b\bar{l} + \sum_{i=1}^{n} b_i l_i & \geq d_2 \\
  \sum_{i=1}^{n} (b a_i + a b_i) l_i & \geq b d_1 + a d_2 - ab
\end{align*}
\]
(cancellation)

\[
\begin{align*}
  \sum_{i=1}^{n} a_i l_i & \geq d \\
  \sum_{i=1}^{n} \min(a_i, d) l_i & \geq d
\end{align*}
\]
(saturation)
Generalized Resolution

The generalized resolution proof system [Hooker, 1988] is used as the counterpart of the resolution proof system in PB solvers such as Sat4j

$$a l + \sum_{i=1}^{n} a_i l_i \geq d_1 \quad b \bar{l} + \sum_{i=1}^{n} b_i l_i \geq d_2$$

(cancellation)

$$\sum_{i=1}^{n} (b a_i + a b_i) l_i \geq b d_1 + a d_2 - a b$$

$$\sum_{i=1}^{n} a_i l_i \geq d$$

(saturation)

$$\sum_{i=1}^{n} \min(a_i, d) l_i \geq d$$
Generalized Resolution

The generalized resolution proof system [Hooker, 1988] is used as the counterpart of the resolution proof system in PB solvers such as Sat4j

\[
\frac{al + \sum_{i=1}^{n} a_i l_i \geq d_1}{\sum_{i=1}^{n} (ba_i + ab_i)l_i \geq b d_1 + a d_2 - ab} \quad \text{(cancellation)}
\]

\[
\frac{\sum_{i=1}^{n} a_i l_i \geq d}{\sum_{i=1}^{n} \min(a_i, d) l_i \geq d} \quad \text{(saturation)}
\]
The generalized resolution proof system [Hooker, 1988] is used as the counterpart of the resolution proof system in PB solvers such as Sat4j.

\[
\begin{align*}
\frac{a_l + \sum_{i=1}^{n} a_i l_i \geq d_1}{\sum_{i=1}^{n} (ba_i + ab_i) l_i \geq bd_1 + ad_2 - ab} \quad \text{(cancellation)} \\
\frac{\sum_{i=1}^{n} a_i l_i \geq d}{\sum_{i=1}^{n} \min(a_i, d) l_i \geq d} \quad \text{(saturation)}
\end{align*}
\]

These two rules are used during conflict analysis to learn new constraints, but have very different properties compared to the resolution proof system used in classical SAT solvers.
Preserving Conflicts
Suppose that we have the following constraints:

\[6\bar{b} + 6c + 4e + f + g + h \geq 7\]  \[5a + 4b + c + d \geq 6\]
Analyzing Conflicts

Suppose that we have the following constraints:

\[ 6b + 6c + 4e + f + g + h \geq 7 \]
\[ 5a + 4b + c + d \geq 6 \]
Suppose that we have the following constraints:

\[ 6b + 6c + 4e + f + g + h \geq 7 \]
\[ 5a + 4b + c + d \geq 6 \]
Suppose that we have the following constraints:

\[ 6b + 6c + 4e + f + g + h \geq 7 \]
\[ 5a + 4b + c + d \geq 6 \]
Suppose that we have the following constraints:

\[ 6\bar{b} + 6c + 4e + f + g + h \geq 7 \quad \text{and} \quad 5a + 4b + c + d \geq 6 \]

(reason for \(\bar{b}\))
Analyzing Conflicts

Suppose that we have the following constraints:

\[ 6\bar{b} + 6c + 4e + f + g + h \geq 7 \quad \text{and} \quad 5a + 4b + c + d \geq 6 \]

(reason for \( \bar{b} \)) (conflict)
Analyzing Conflicts

Suppose that we have the following constraints:

\[ 6\bar{b} + 6c + 4e + f + g + h \geq 7 \]

(\text{reason for } \bar{b})

\[ 5a + 4b + c + d \geq 6 \]

(\text{conflict})

This conflict is analyzed by applying the cancellation rule as follows:

\[
\begin{align*}
6\bar{b} + 6c + 4e + f + g + h & \geq 7 \\
5a + 4b + c + d & \geq 6 \\
\hline
15a + 15c + 8e + 3d + 2f + 2g + 2h & \geq 20
\end{align*}
\]
Analyzing Conflicts

Suppose that we have the following constraints:

\[
6\bar{b} + 6c + 4e + f + g + h \geq 7 \quad 5a + 4b + c + d \geq 6
\]

(reason for \(\bar{b}\))

This conflict is analyzed by applying the cancellation rule as follows:

\[
\frac{6\bar{b} + 6c + 4e + f + g + h \geq 7 \quad 5a + 4b + c + d \geq 6}{15a + 15c + 8e + 3d + 2f + 2g + 2h \geq 20}
\]

The constraint we obtain here is no longer conflicting!
To preserve the conflict, the *weakening* rule must be used:

\[
\frac{a l + \sum_{i=1}^{n} a_i l_i \geq d}{\sum_{i=1}^{n} a_i l_i \geq d-a} \quad \text{(weakening)}
\]
To preserve the conflict, the weakening rule must be used:

\[
\frac{al + \sum_{i=1}^{n} a_i l_i \geq d}{\sum_{i=1}^{n} a_i l_i \geq d-a} \quad \text{(weakening)}
\]
To preserve the conflict, the *weakening* rule must be used:

\[
\frac{al + \sum_{i=1}^{n} a_i l_i \geq d}{\sum_{i=1}^{n} a_i l_i \geq d - a} \quad \text{(weakening)}
\]
To preserve the conflict, the weakening rule must be used:

\[
\frac{al + \sum_{i=1}^{n} a_\i \l_i \geq d}{\sum_{i=1}^{n} a_\i \l_i \geq d - a}
\] (weakening)

Weakening can be applied in many different ways!
The Original Weakening Strategy

The original approach [Dixon, 2002; Chai & Kuehlmann, 2003] successively weakens away literals from the reason, until the saturation rule guarantees to derive a conflicting constraint.
The original approach [Dixon, 2002; Chai & Kuehlmann, 2003] successively weakens away literals from the reason, until the saturation rule guarantees to derive a conflicting constraint.

To check whether the constraint we obtain is conflictual, we can use the slack of the constraints:

\[
\text{slack} \left( \sum_{i=1}^{n} a_i l_i \geq d \right) = \left( \sum_{i=1, l_i \neq 0}^{n} a_i \right) - d
\]
The Original Weakening Strategy

The original approach [Dixon, 2002; Chai & Kuehlmann, 2003] successively weakens away literals from the reason, until the saturation rule guarantees to derive a conflicting constraint.

To check whether the constraint we obtain is conflictual, we can use the slack of the constraints

\[
\text{slack} \left( \sum_{i=1}^{n} a_i l_i \geq d \right) = \left( \sum_{i=1, l_i \neq 0}^{n} a_i \right) - d
\]

The slack is subadditive: the slack of a constraint obtained by applying the cancellation rule is at most equal to the sum of the slacks of the two original constraints.
The Original Weakening Strategy

The original approach [Dixon, 2002; Chai & Kuehlmann, 2003] successively weakens away literals from the reason, until the saturation rule guarantees to derive a conflicting constraint.

To check whether the constraint we obtain is conflictual, we can use the slack of the constraints

$$\text{slack}\left(\sum_{i=1}^{n} a_i l_i \geq d\right) = \left(\sum_{i=1, l_i \neq 0}^{n} a_i\right) - d$$

The slack is subadditive: the slack of a constraint obtained by applying the cancellation rule is at most equal to the sum of the slacks of the two original constraints.

This property gives an upper-bound of the slack of the produced constraint without actually computing the cancellation, its cost is not negligible as the operation must be repeated multiple times.
An Important Property

In some cases, we do not need to estimate the slack, as we are sure that the constraint that will be derived will be conflicting.

As soon as the coefficient of the literal to cancel is equal to 1 in at least one of the constraints, the derived constraint is guaranteed to be conflicting [Dixon, 2004]
An Important Property

In some cases, we do not need to estimate the slack, as we are sure that the constraint that will be derived will be conflicting.

As soon as the coefficient of the literal to cancel is equal to 1 in at least one of the constraints, the derived constraint is guaranteed to be conflicting [Dixon, 2004]

This property is true if the coefficient of the constraint is 1 in the constraint encountered during conflict analysis, or if we apply some operations that make it equal to 1.
In some cases, we do not need to estimate the slack, as we are sure that the constraint that will be derived will be conflicting.

As soon as the coefficient of the literal to cancel is **equal to 1 in at least one of the constraints**, the derived constraint is **guaranteed to be conflicting** [Dixon, 2004]

This property is true if the coefficient of the constraint is 1 in the constraint encountered during conflict analysis, or if we apply some operations that **make it equal to 1**

Different weakening strategies allow to do so
The weakening strategies that follow are not applied at each derivation step during conflict analysis, but only when the coefficient of the pivot is **not equal to 1** in both the conflict and in the reason, as otherwise we are sure that the conflict will be preserved by the previous property.
Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants.
Weakening Ineffective Literals

Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
3\bar{a} + 3\bar{b} + c + d + e \geq 6
\]

\[
\bar{b} + c \geq 1
\]
Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
3\bar{a} + 3\bar{b} + c + d + e \geq 6
\]

\[
\begin{align*}
3\bar{b} + c & \geq 1 \\
\bar{b} + c & \geq 1
\end{align*}
\]
Weakening Ineffective Literals

Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
3\bar{a} + 3\bar{b} + c + d + e \geq 6
\]
\[
3\bar{b} + c \geq 1
\]
\[
\bar{b} + c \geq 1
\]
Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
\begin{align*}
3\bar{a} + 3\bar{b} + c + d + e & \geq 6 \\
3\bar{b} + c & \geq 1 \\
\bar{b} + c & \geq 1
\end{align*}
\]
Weakening Ineffective Literals

Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
\begin{align*}
3\bar{a} + 3\bar{b} + c + d + e & \geq 6 \\
\frac{3\bar{b} + c \geq 1}{\bar{b} + c \geq 1}
\end{align*}
\]

\[
\begin{align*}
2a + b + c + f & \geq 2 \\
\frac{2a + b + f \geq 1}{a + b + f \geq 1}
\end{align*}
\]
Some literals **may not play a role** in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
\begin{align*}
3\bar{a} + 3\bar{b} + c + d + e & \geq 6 \\
\frac{3\bar{b} + c \geq 1}{3\bar{b} + c} \\
\bar{b} + c & \geq 1
\end{align*}
\]

\[
\begin{align*}
2a + b + c + f & \geq 2 \\
\frac{2a + b + f \geq 1}{2a + b + f} \\
a + b + f & \geq 1
\end{align*}
\]
Weakening Ineffective Literals

Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants.

\[
\begin{array}{c}
3\overline{a} + 3\overline{b} + c + d + e \geq 6 \\
\overline{3b} + c \geq 1 \\
\overline{b} + c \geq 1 \\
\end{array}
\quad
\begin{array}{c}
2a + b + c + f \geq 2 \\
2a + b + f \geq 1 \\
a + b + f \geq 1 \\
\end{array}
\]
Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
\begin{align*}
3\overline{a} + 3\overline{b} + c + d + e & \geq 6 \\
\overline{3b} + c & \geq 1 \\
\overline{b} + c & \geq 1
\end{align*}
\]

\[
\begin{align*}
2a + b + c + f & \geq 2 \\
2a + b + f & \geq 1 \\
a + b + f & \geq 1
\end{align*}
\]
Weakening Ineffective Literals

Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[ 3\overline{a} + 3\overline{b} + c + d + e \geq 6 \]
\[ \overline{3b} + c \geq 1 \]
\[ \overline{b} + c \geq 1 \]

\[ 2a + b + c + f \geq 2 \]
\[ 2a + b + f \geq 1 \]
\[ a + b + f \geq 1 \]

This strategy is equivalent to that used by solvers such as SATIRE or Sat4j-Resolution to lazily infer clauses to use resolution based reasoning.
Weakening Ineffective Literals

Some literals may not play a role in the conflict or the propagation: it is thus possible to weaken them away while preserving invariants

\[
\begin{align*}
3\bar{a} + 3\bar{b} + c + d + e &\geq 6 \\
3\bar{b} + c &\geq 1 \\
\bar{b} + c &\geq 1
\end{align*}
\]

\[
\begin{align*}
2a + b + c + f &\geq 2 \\
2a + b + f &\geq 1 \\
a + b + f &\geq 1
\end{align*}
\]

This strategy is equivalent to that used by solvers such as SATIRE or Sat4j-Resolution to lazily infer clauses to use resolution based reasoning.

We propose here to apply it on one side of the cancellation, to infer stronger constraints and preserve PB reasoning.
Weakening Ineffective Literals (Experiments)

![Graph showing the comparison of different algorithms: VBS, Sat4j-RoundingSat, Sat4j-WeakenIneffective (both), Sat4j-WeakenIneffective (conflict), Sat4j-WeakenIneffective (reason), and Sat4j-GeneralizedResolution. The x-axis represents the number of instances, and the y-axis represents the time in seconds. The graph visually demonstrates the performance of these algorithms across different numbers of instances.]
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals.
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals

\[
\frac{8a + 7b + 7c + 2d + 2e + f \geq 11}{7b + 7c + 2d + 2e \geq 2}\]
\[
\frac{7b + 7c + 2d + 2e \geq 2}{b + c + d + e \geq 1}\]
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7b + 7c + 2d + 2e & \geq 2 \\
b + c + d + e & \geq 1
\end{align*}
\]
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7b + 7c + 2d + 2e & \geq 2 \\
\quad b + c + d + e & \geq 1
\end{align*}
\]
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals

\[
\frac{8a + 7b + 7c + 2d + 2e + f}{7b + 7c + 2d + 2e} \geq 2
\]

\[
\frac{b + c + d + e}{1}
\]
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals

\[
\frac{8a + 7b + 7c + 2d + 2e + f \geq 11}{7b + 7c + 2d + 2e \geq 2} \quad b + c + d + e \geq 1
\]

*RoundingSat* applies this operation on both sides of the cancellation
In *RoundingSat* [Elffers & Nordström, 2018], the coefficient is rounded to one thanks to the division rule, applied after having weakened away some unfalsified literals

\[
\frac{8a + 7b + 7c + 2d + 2e + f \geq 11}{7b + 7c + 2d + 2e \geq 2} \quad \frac{b + c + d + e \geq 1}
\]

*RoundingSat* applies this operation on both sides of the cancellation.

*Once again, we propose here to apply this operation on only one side of the cancellation*
Weakening and Division (Experiments)
Another possibility is to consider a variant of the weakening rule, known as partial weakening.

\[
al + \sum_{i=1}^{n} a_i l_i \geq d \\
(k - k)l + \sum_{i=1}^{n} a_i l_i \geq d - k
\]

In general, this rule allows to derive stronger constraints than with the weakening rule.
Another possibility is to consider a variant of the weakening rule, known as partial weakening.

\[ al + \sum_{i=1}^{n} a_i l_i \geq d \quad k \in \mathbb{N} \quad 0 < k \leq a \quad (\text{partial weakening}) \]
Another possibility is to consider a variant of the weakening rule, known as partial weakening.

\[ al + \sum_{i=1}^{n} a_i l_i \geq d \quad k \in \mathbb{N} \quad 0 < k \leq a \]  
\[ (a - k)l + \sum_{i=1}^{n} a_i l_i \geq d - k \]  
(partial weakening)
Another possibility is to consider a variant of the weakening rule, known as partial weakening.

\[ al + \sum_{i=1}^{n} a_i l_i \geq d \]

\[ k \in \mathbb{N} \quad 0 < k \leq a \]

\[ (a - k) l + \sum_{i=1}^{n} a_i l_i \geq d - k \]

(partial weakening)

In general, this rule allows to derive **stronger constraints than with the weakening rule.**
Considering a similar idea to that of *RoundingSat*, we propose to use **partial weakening** instead of weakening.
Partial Weakening and Division

Considering a similar idea to that of *RoundingSat*, we propose to use partial weakening instead of weakening

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7a + 7b + 7c + 2d + 2e & \geq 9 \\
a + b + c + d + e & \geq 2
\end{align*}
\]
Considering a similar idea to that of *RoundingSat*, we propose to use **partial weakening** instead of weakening

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7a + 7b + 7c + 2d + 2e & \geq 9 \\
a + b + c + d + e & \geq 2
\end{align*}
\]
Partial Weakening and Division

Considering a similar idea to that of *RoundingSat*, we propose to use partial weakening instead of weakening

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7a + 7b + 7c + 2d + 2e & \geq 9 \\
a + b + c + d + e & \geq 2
\end{align*}
\]
Partial Weakening and Division

Considering a similar idea to that of *RoundingSat*, we propose to use partial weakening instead of weakening

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7a + 7b + 7c + 2d + 2e & \geq 9 \\
a + b + c + d + e & \geq 2
\end{align*}
\]
Considering a similar idea to that of *RoundingSat*, we propose to use partial weakening instead of weakening

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7a + 7b + 7c + 2d + 2e & \geq 9 \\
a + b + c + d + e & \geq 2
\end{align*}
\]

Observe that the constraint obtained here is stronger than the clause \( b + c + d + e \geq 1 \) derived by *RoundingSat*
Partial Weakening and Division

Considering a similar idea to that of \textit{RoundingSat}, we propose to use \textit{partial weakening} instead of weakening

\[
\begin{align*}
8a + 7b + 7c + 2d + 2e + f & \geq 11 \\
7a + 7b + 7c + 2d + 2e & \geq 9 \\
a + b + c + d + e & \geq 2
\end{align*}
\]

Observe that the constraint obtained here is \textit{stronger} than the clause \(b + c + d + e \geq 1\) derived by \textit{RoundingSat}

\textit{This operation may be applied on either one or both sides of the cancellation}
Partial Weakening and Division (Experiments)
Complete Experiments

The image shows a graph with the x-axis labeled as "Number of instances" ranging from 3500 to 4000, and the y-axis labeled as "Time (s)" ranging from 0 to 1250. The graph compares various algorithms, including:

- VBS
- Sat4j−PartialRoundingSat (both)
- Sat4j−RoundingSat
- Sat4j−WeakenIneffective (both)
- Sat4j−WeakenIneffective (conflict)
- Sat4j−PartialRoundingSat (conflict)
- Sat4j−RoundingSat (conflict)
- Sat4j−WeakenIneffective (reason)
- Sat4j−RoundingSat (reason)
- Sat4j−PartialRoundingSat (reason)
- Sat4j−MultiplyAndWeaken
- Sat4j−GeneralizedResolution
Choosing Decision Variables
A Conflict Analysis

Suppose that we have the following constraints:

\[ 3\bar{a}(?@?) + 3\bar{f}(?@?) + d(?@?) + e(?@?) \geq 5 \]
\[ 6a(?@?) + 3b(?@?) + 3c(?@?) + 3d(?@?) + 3f(?@?) \geq 9 \]
A Conflict Analysis

Suppose that we have the following constraints:

\[
3\bar{a}(\oplus?) + 3\bar{f}(\oplus?) + d(\oplus?) + e(\oplus?) \geq 5 \\
6a(\oplus?) + 3b(1\oplus) + 3c(\oplus?) + 3d(\oplus?) + 3f(\oplus?) \geq 9
\]
A Conflict Analysis

Suppose that we have the following constraints:

\[ 3\overline{a}(\text{@}) + 3\overline{f}(\text{@}) + d(\text{@}) + e(\text{@}) \geq 5 \]
\[ 6a(\text{@}) + 3b(1\text{@}0) + 3c(0\text{@}2) + 3d(\text{@}) + 3f(\text{@}) \geq 9 \]
A Conflict Analysis

Suppose that we have the following constraints:

\[
3\bar{a}(?@?) + 3\bar{f}(?@?) + d(0@3) + e(?@?) \geq 5
\]
\[
6a(?@?) + 3b(1@1) + 3c(0@2) + 3d(?@?) + 3f(?@?) \geq 9
\]
A Conflict Analysis

Suppose that we have the following constraints:

\[
3\bar{a}(1@3) + 3\bar{f}(1@3) + d(0@3) + e(?@?) \geq 5 \\
6a(0@3) + 3b(1@1) + 3c(0@2) + 3d(?@?) + 3f(0@3) \geq 9
\]
Suppose that we have the following constraints:

\[ 3\overline{a}(1@3) + 3\overline{f}(1@3) + d(0@3) + e(?@?) \geq 5 \]
\[ 6a(0@3) + 3b(1@1) + 3c(0@2) + 3d(?@?) + 3f(0@3) \geq 9 \]

We now apply the cancellation rule between these two constraints:

\[ \frac{3\overline{a} + 3\overline{f} + d + e \geq 5}{3a(0@3) + 3b(1@1) + 3c(0@2) + 2\overline{d}(1@3) + e(?@?) \geq 7} \]
\[ 6a + 3b + 3c + 3d + 3f \geq 9 \]
A Conflict Analysis

Suppose that we have the following constraints:

\[
3\bar{a}(1\oplus 3) + 3\bar{f}(1\oplus 3) + d(0\oplus 3) + e(0\oplus 3) \geq 5 \\
6a(0\oplus 3) + 3b(1\oplus 1) + 3c(0\oplus 2) + 3d(0\oplus 2) + 3f(0\oplus 3) \geq 9
\]

We now apply the cancellation rule between these two constraints:

\[
\begin{align*}
3\bar{a} + 3\bar{f} + d + e & \geq 5 \\
6a + 3b + 3c + 3d + 3f & \geq 9 \\
\hline
3a(0\oplus 3) + 3b(1\oplus 1) + 3c(0\oplus 2) + 2d(0\oplus 2) + e(0\oplus 3) & \geq 7
\end{align*}
\]
A Conflict Analysis

Suppose that we have the following constraints:

\[
3\bar{a}(1@3) + 3\bar{f}(1@3) + d(0@3) + e(?@?) \geq 5
\]
\[
6a(0@3) + 3b(1@1) + 3c(0@2) + 3d(?@?) + 3f(0@3) \geq 9
\]

We now apply the cancellation rule between these two constraints:

\[
\begin{align*}
3\bar{a} + 3\bar{f} + d + e & \geq 5 \\
6a + 3b + 3c + 3d + 3f & \geq 9
\end{align*}
\]
\[
3a(?@?) + 3b(1@1) + 3c(0@2) + 2d(?@?) + e(?@?) \geq 7
\]

The PB constraints involved in this conflict analysis have very different properties compared to clauses!
All variables encountered during conflict analysis are bumped.
All variables encountered during conflict analysis are **bumped**.

This is the case for all the variables appearing in the previous **reason**:

\[ 3\overline{a} + 3\overline{f} + d + e \geq 5 \]
All variables encountered during conflict analysis are **bumped**

This is the case for all the variables appearing in the previous **reason**:

\[ 3\bar{a} + 3\bar{f} + d + e \geq 5 \]

*This means that the scores of the variables \(a, f, d\) and \(e\) are **incremented***
A first approach for adapting VSIDS to PB constraints has been proposed in [Dixon, 2004], but it only takes into account the original cardinality constraints, by incrementing the score of each variable by the value of the degree

\[ a + b + c \geq 2 \]
A first approach for adapting VSIDS to PB constraints has been proposed in [Dixon, 2004], but it only takes into account the original cardinality constraints, by incrementing the score of each variable by the value of the degree

$$a + b + c \geq 2$$

However, this approach does not take into account the coefficients in a PB constraint, contrary to the implementation proposed in Pueblo [Sheini and Sakallah, 2006], which increments the score of the variables by the value of the coefficient of a variable divided by the degree (e.g., 3/5 for $a$ in the reason below)

$$3\overline{a} + 3\overline{f} + d + e \geq 5$$
Considering again the constraint we used as a reason before

\[ 3\bar{a} + 3\bar{f} + d + e \geq 5 \]

We propose to take its coefficients into account with 3 other strategies:
Considering again the constraint we used as a reason before

\[3a + 3f + d + e \geq 5\]

We propose to take its coefficients into account with 3 other strategies:

- **bump-degree**: the score of each variable is incremented by the degree of the constraint (5 for all variables)
Considering again the constraint we used as a reason before

\[ 3\overline{a} + 3\overline{f} + d + e \geq 5 \]

We propose to take its coefficients into account with 3 other strategies:

- **bump-degree**: the score of each variable is incremented by the **degree** of the constraint (5 for all variables)

- **bump-coefficient**: the score of each variable is incremented by their **coefficients** in the constraint (3 for \( a \) and \( f \))
Considering again the constraint we used as a reason before

\[3\bar{a} + 3\bar{f} + d + e \geq 5\]

We propose to take its coefficients into account with 3 other strategies:

- **bump-degree**: the score of each variable is incremented by the degree of the constraint (5 for all variables)

- **bump-coefficient**: the score of each variable is incremented by their coefficients in the constraint (3 for \(a\) and \(f\))

- **bump-ratio-degree-coefficient**: the score of each variable is incremented by the ratio of the degree by their coefficient in the constraint (5/3 for \(a\) and \(f\))
We can also take into account the current assignment when bumping variables

\[ 3\bar{a} + 3\bar{f} + d + e \geq 5 \]
We can also take into account the current assignment when bumping variables

\[3\overline{a} + 3\overline{f} + d + e \geq 5\]

- bump-assigned increments the score of each assigned variable \((a, f\text{ and } d)\)
We can also take into account the current assignment when bumping variables

\[3\bar{a} + 3\bar{f} + d + e \geq 5\]

- bump-assigned increments the score of each assigned variable \((a, f\) and \(d)\)
- bump-falsified increments the score of each falsified variable \((d)\)
We can also take into account the current assignment when bumping variables

\[ 3\bar{a} + 3\bar{f} + d + e \geq 5 \]

- **bump-assigned** increments the score of each assigned variable \((a, f\) and \(d)\)
- **bump-falsified** increments the score of each falsified variable \((d)\)
- **bump-falsified-propagated** increments the score of each falsified and propagated variable \((a, f\) and \(d)\)
We can also take into account the current assignment when bumping variables

\[3\bar{a} + 3\bar{f} + d + e \geq 5\]

- bump-assigned increments the score of each assigned variable (\(a, f\) and \(d\))
- bump-falsified increments the score of each falsified variable (\(d\))
- bump-falsified-propagated increments the score of each falsified and propagated variable (\(a, f\) and \(d\))
- bump-effective: increments the score of each effective variable (\(d\))
We can also take into account the current assignment when bumping variables

\[ 3\bar{a} + 3\bar{f} + d + e \geq 5 \]

- **bump-assigned** increments the score of each assigned variable \((a, f, \text{ and } d)\)
- **bump-falsified** increments the score of each falsified variable \((d)\)
- **bump-falsified-propagated** increments the score of each falsified and propagated variable \((a, f, \text{ and } d)\)
- **bump-effective** increments the score of each effective variable \((d)\)
- **bump-effective-propagated** increments the score of each effective and propagated variable \((a, f, \text{ and } d)\)
(E)VSIDS: Experiments (Sat4j-GR)

![Graph showing time (s) vs. number of instances for different variables and conditions.

- VBS
- bump-effective
- bump-falsified
- bump-effective-propagated
- bump-falsified-propagated
- bump-assigned
- bump-ratio-coefficient-degree
- bump-default
- bump-coefficient
- bump-degree
- bump-ratio-degree-coefficient]
(E)VSIDS: Experiments (Sat4j-RS)
(E)VSIDS: Experiments (Sat4j-PartialRS)
Learned Constraint Quality
In SAT solvers, evaluating the quality of learned constraints is used to choose which constraints should be deleted and to decide when a restart should be triggered.
In SAT solvers, evaluating the quality of learned constraints is used to choose which constraints should be deleted and to decide when a restart should be triggered.

*The quality measures used by SAT solvers do not take into account the properties of PB constraints*
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.

In the PB case, the length of a constraint does not reflect its strength.
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.

In the PB case, the length of a constraint does not reflect its strength.

However, the size of a PB constraint also takes into account its coefficients.
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.

In the PB case, the length of a constraint does not reflect its strength. However, the size of a PB constraint also takes into account its coefficients.

Consider the constraint we derived in the previous conflict analysis:

$$3a + 3b + 3c + 2d + e \geq 7$$
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.

In the PB case, the length of a constraint does not reflect its strength. However, the size of a PB constraint also takes into account its coefficients.

Consider the constraint we derived in the previous conflict analysis:

$$3a + 3b + 3c + 2\bar{d} + e \geq 7$$

In practice, the coefficients may become very big, which requires the use of arbitrary precision encodings and slows down arithmetic operations.
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.

In the PB case, the length of a constraint does not reflect its strength.

However, the size of a PB constraint also takes into account its coefficients.

Consider the constraint we derived in the previous conflict analysis:

\[ 3a + 3b + 3c + 2d + e \geq 7 \]

In practice, the coefficients may become very big, which requires the use of arbitrary precision encodings and slows down arithmetic operations.

We consider quality measures based on the value or the size of the degree of the constraints: the lower the degree, the better the constraint.
In SAT solvers, the size of a clause is a naive measure of its quality: the longer the clause, the lower its strength.

In the PB case, the length of a constraint does not reflect its strength.

However, the size of a PB constraint also takes into account its coefficients.

Consider the constraint we derived in the previous conflict analysis:

\[3a + 3b + 3c + 2d + e \geq 7\]

In practice, the coefficients may become very big, which requires the use of arbitrary precision encodings and slows down arithmetic operations.

We consider quality measures based on the value or the size of the degree of the constraints: the lower the degree, the better the constraint.
Another indicator that we have for evaluating the quality of a constraint is to estimate its strength with its slack

$$3a + 3b + 3c + 2d + e \geq 7$$

In this case, we prefer to consider the \textit{absolute} slack of the constraint, independently of the current assignment: in this example, it is equal to 5 (while, under the current assignment, it is equal to −1)

\begin{quote}
We consider quality measures based on the value of the \textit{slack} of the constraints: \textit{the lower the slack, the better the constraint}
\end{quote}
In SAT solvers, the **Literal Block Distance (LBD)** measures the quality of clauses by the number of decision levels appearing in this clause.
In SAT solvers, the **Literal Block Distance (LBD)** measures the quality of clauses by the number of decision levels appearing in this clause.

Consider again the constraint we derived previously:

\[ 3a(0@3) + 3b(1@1) + 3c(0@2) + 2\overline{d}(1@3) + e(?@?) \geq 7 \]
Quality of Learned Constraint: Assignments (LBD)

In SAT solvers, the **Literal Block Distance (LBD)** measures the quality of clauses by the number of decision levels appearing in this clause.

Consider again the constraint we derived previously:

$$3a(0@3) + 3b(1@1) + 3c(0@2) + 2\overline{d}(1@3) + e(?@?) \geq 7$$

There are satisfied and unassigned literals in this constraint!
In SAT solvers, the **Literal Block Distance (LBD)** measures the quality of clauses by the number of decision levels appearing in this clause.

Consider again the constraint we derived previously:

\[
3a(0\oplus3) + 3b(1\oplus1) + 3c(0\oplus2) + 2\bar{d}(1\oplus3) + e(\oplus?) \geq 7
\]

There are satisfied and unassigned literals in this constraint!

We thus introduce 4 new definitions of LBD:

- $\text{lbd-a}$: the LBD is computed over assigned literals only
- $\text{lbd-s}$: the LBD is computed over assigned literals, and unassigned literals are considered assigned at the same (dummy) decision level
- $\text{lbd-d}$: the LBD is computed over assigned literals, and unassigned literals are considered assigned at different (dummy) decision levels
- $\text{lbd-f}$: the LBD is computed over falsified literals only
- $\text{lbd-e}$: the LBD is computed over effective literals only
Deleting constraints is required by SAT solvers to limit the memory usage and to prevent unit propagation from slowing down.
Deleting constraints is required by SAT solvers to limit the memory usage and to prevent unit propagation from slowing down.

This is also true, but to a lesser extent, for PB solvers.
Deleting constraints is required by SAT solvers to limit the memory usage and to prevent unit propagation from slowing down.

This is also true, but to a lesser extent, for PB solvers.

The constraints to delete are those having a bad score w.r.t. the quality measure used in the solver.
Deleting constraints is required by SAT solvers to limit the memory usage and to prevent unit propagation from slowing down.

This is also true, but to a lesser extent, for PB solvers.

The constraints to delete are those having a bad score w.r.t. the quality measure used in the solver.

We thus introduce the following deletion strategies:

- delete-degree
- delete-degree-size
- delete-slack
- delete-lbd-a
- delete-lbd-s
- delete-lbd-d
- delete-lbd-f
- delete-lbd-e
Learned Constraint Deletion: Experiments (Sat4j-GR)
Restarting allows to forget all decisions made by the solver, so as to avoid being stuck in a subpart of the search space.
Restarting allows to forget all decisions made by the solver, so as to avoid being stuck in a subpart of the search space.

Following Glucose's approach, we consider adaptive restarts based on the quality of recently learned constraints.
Quality of Learned Constraint: Restarts

Restarting allows to forget all decisions made by the solver, so as to avoid being stuck in a subpart of the search space.

Following Glucose’s approach, we consider adaptive restarts based on the quality of recently learned constraints.

Whenever the most recent constraints are of poor quality compared to all the others, a restart is performed.
Quality of Learned Constraint: Restarts

**Restarting** allows to forget all decisions made by the solver, so as to avoid being stuck in a subpart of the search space.

Following *Glucose*’s approach, we consider **adaptive restarts** based on the quality of recently learned constraints.

*Whenever the most recent constraints are of poor quality compared to all the others, a restart is performed.*

We thus introduce the following restart strategies:

- restart-degree
- restart-degree-size
- restart-slack
- restart-lbd-a
- restart-lbd-s
- restart-lbd-d
- restart-lbd-f
- restart-lbd-e
Restarts: Experiments (Sat4j-GR)
Repeats: Experiments (Sat4j-RS)
Combining the Best Strategies
In Sat4j-GeneralizedResolution, the best strategies are

- bump-falsified
- delete-lbd-s
- restart-degree

*Let us combine all these strategies!*
Combining the Best Strategies: Sat4j-GR (Experiments)
Combining the Best Strategies: Sat4j-RS

In Sat4j-RoundingSat, the best strategies are

- bump-assigned
- delete-slack
- restart-picosat

Let us combine all these strategies!
Combining the Best Strategies: Sat4j-RS (Experiments)

![Graph showing the comparison of different strategies for solving number of instances over time. The graph plots the time (s) on the y-axis and the number of instances on the x-axis. The strategies include VBS, best-combination, delete-slack, restart-delete-lbd-d, bump-assigned, restart-picsoat, and default. The graph illustrates the performance of each strategy across various instance sizes.](image)
In Sat4j-PartialRoundingSat, the best strategies are

- bump-assigned
- delete-degree-size
- restart-picosat

*Let us combine all these strategies!*
Combining the Best Strategies: Sat4j-PartialRS (Experiments)
Combining the Best Strategies: Complete Overview

![Graph showing the performance comparison of different SAT solvers and strategies.](image)

- **VBS**
- **best-combination** (Sat4j-PartialRoundingSat)
- **best-combination** (Sat4j-RoundingSat)
- **best-combination** (Sat4j-GeneralizedResolution)
- **default** (Sat4j-PartialRoundingSat)
- **default** (Sat4j-RoundingSat)
- **default** (Sat4j-GeneralizedResolution)
Conclusion and Perspectives
Conclusion

- CDCL in PB solvers requires a particular attention to preserve its properties compared to SAT solvers
- Different weakening strategies may be applied to preserve conflicts
- Bumping variables works better when considering the current assignment
- Considering the coefficients to evaluate the quality of a learned PB constraint provides a quite accurate measure
VBS
RoundingSat
Sat4j–Both
Sat4j–Resolution
Sat4j–PartialRoundingSat
(best–combination)
Sat4j–RoundingSat
(best–combination)
Sat4j–GeneralizedResolution
(best–combination)
Sat4j–PartialRoundingSat
(default)
Sat4j–RoundingSat
(default)
Sat4j–GeneralizedResolution
(default)
Perspectives

- Consider more specifically the impact of the weakening rule on either the conflict or the reason side of the cancellation rule
- Find better tradeoffs to combine the different weakening strategies
- Find better extension or combinations of the presented CDCL strategies
- Consider all the presented strategies on optimization problems
Tuning Sat4j PB Solvers for Decision Problems

Romain Wallon
Zoom Seminar – August 28th, 2020
CRIL, Univ Artois & CNRS