PBSugar
Compiling Pseudo-Boolean Constraints to SAT with Order Encoding

Naoyuki Tamura, Mutsunori Banbara, Takehide Soh
Kobe University

July 26, 2013
3rd CSPSAT2 Meeting in Sapporo
We implemented a Pseudo-Boolean (PB for short) solver named PBSugar by applying the order encoding technique to PB constraints.

Contents

- Order encoding
  - Original version
  - Optimized version
- BC constraints
  - Encoding of BC constraints by the order encoding
- PB constraints
  - Encoding of PB constraints by the order encoding
- PBSugar Implementation
  - Experimental Results
- Conclusion
In [Tamura et al. 2006], we proposed the encoding to be applied for finite linear CSPs (Constraint Satisfaction Problems).

- It is implemented in Sugar solver which won in Global categories of 2008 and 2009 CSP solver competitions.
- Boolean variable $p(x \geq c)$ (alternatively $p(x \leq c)$) is assigned for each integer variable $x$ and each domain value $c$.
- Bounds consistency is maintained by Unit Propagation.
- It is the only existing encoding translating tractable CSP to tractable SAT [Petke and Jeavons 2011].
- It showed a good performance for a wide range of problems.

---

1[Crawford and Baker 1994] [Bailleux et al. 2003] [Ansótegui and Manyà 2004] [Gent and Nightingale 2004] [Ohrimenko et al. 2009] [Metodi et al. 2011]

First, we introduce the order encoding of the original version [Tamura et al. 2006].

Let $x$ be an integer variable where $\text{Dom}(x) = \{l..u\}$.

### Boolean variables used for $x \in \{l..u\}$

We introduce $u - l - 1$ Boolean variables for $x$:

$$p(x \geq c) \quad (\text{for each } c \in \{l+1..u\})$$

### Axiom clauses used for $x \in \{l..u\}$

The axiom clauses enforce $x$ to take exactly one value in $\{l..u\}$.

$$p(x \geq c) \lor \neg p(x \geq c + 1) \quad (\text{for each } c \in \{l+1..u-1\})$$

---

$^3p(x \leq c)$ is used instead in [Tamura et al. 2006] and Sugar system.
Let $C$ be a linear arithmetic constraint $\sum_{i=1}^{n} a_i x_i \geq c$ where each $a_i$ is a non-zero integer constant, $c$ is an integer constant, and each $x_i$ is an integer variable with the domain $Dom(x_i)$.

Original Order Encoding of $\sum_{i=1}^{n} a_i x_i \geq c$

\[
\sum_{i=1}^{n} a_i x_i \geq c \iff \bigwedge_{c_i} \bigvee_{i=1}^{n} (a_i x_i \geq c_i)^\# 
\]

- Parameters $c_i$’s range over integers satisfying $\sum_{i=1}^{n} c_i = c + n - 1$ and $\inf(a_i x_i) \leq c_i \leq \sup(a_i x_i) + 1$.
- The translation $(ax \geq c)^\#$ is defined by:

\[
(ax \geq c)^\# = \begin{cases} 
  x \geq \lceil c/a \rceil & \text{ (when } a > 0) \\
  -(x \geq \lfloor c/a \rfloor + 1) & \text{ (when } a < 0) 
\end{cases}
\]
Order Encoding: Original Version (3/3)

Straightforward implementation of the original version may generate redundant clauses.

Consider $2x_1 + 3x_2 \geq 20$ where $\text{Dom}(x_1) = \text{Dom}(x_2) = \{0..8\}$. The parameters $c_1$ and $c_2$ range over $\{0..17\}$ and $\{0..25\}$ respectively satisfying $c_1 + c_2 = 21$.

- We obtain 18 clauses including one valid clause.
- Some clauses are appeared twice:
  \[
  (2x_1 \geq 1) \lor (3x_2 \geq 20) \longrightarrow p(x_1 \geq 1) \lor p(x_2 \geq 7)
  \]
  \[
  (2x_1 \geq 2) \lor (3x_2 \geq 19) \longrightarrow p(x_1 \geq 1) \lor p(x_2 \geq 7)
  \]
- Some clauses can be derived from others:
  \[
  (2x_1 \geq 4) \lor (3x_2 \geq 17) \longrightarrow p(x_1 \geq 2) \lor p(x_2 \geq 6)
  \]
  \[
  (2x_1 \geq 5) \lor (3x_2 \geq 16) \longrightarrow p(x_1 \geq 3) \lor p(x_2 \geq 6)
  \]
Now, we revise the original version.

First, we allow non-contiguous domains for variables. Let $x$ be an integer variable where $Dom(x) = \{d_0, d_1, \ldots, d_m\}$ ($d_0 < d_1 < \cdots < d_m$).

**Boolean variables used for $x \in Dom(x)$**

We introduce $m$ Boolean variables:

$$p(x \geq d_j) \quad (\text{for each } j \in \{1..m\})$$

**Axiom clauses used for $x \in Dom(x)$**

The axiom clauses enforce $x$ to have exactly one value in $Dom(x)$.

$$p(x \geq d_j) \lor \neg p(x \geq d_{j+1}) \quad (\text{for each } j \in \{1..m - 1\})$$
Order Encoding: Optimized Version (2/4)

We can show the following proposition.

**Proposition**

\[
\sum_{i=1}^{n} a_i x_i \geq c \iff \\
\begin{cases}
(x_1 \geq \lceil c/a_1 \rceil) \\
\neg(x_1 \geq \lceil c/a_1 \rceil + 1)
\end{cases}
\]

\[
\bigwedge_{d \in \text{Dom}(x_1)} \left((x_1 \geq d + 1) \lor \sum_{i=2}^{n} a_i x_i \geq c - a_1 d\right)
\]

\[
\bigwedge_{d \in \text{Dom}(x_1)} \left(\neg(x_1 \geq d) \lor \sum_{i=2}^{n} a_i x_i \geq c - a_1 d\right)
\]

- Now, the big conjunction iterates over the values of \( x_i \) instead of \( a_i x_i \) in the original version.
We can derive a recursive encoding algorithm from the proposition. However, there remains some room for further optimization.

- First, by sorting the coefficients in the descending order, the number of redundant (derived) clauses will be reduced.

- Second, we can remove clauses when the recursive part \( \sum_{i=2}^{n} a_i x_i \geq c - a_1 d \) is valid.

- Third, we can remove the literal when the recursive part is inconsistent. Also we only need one clause containing such part because other clauses are derived from it.

These ideas have been implemented in Sugar solver.
Consider the same example again: \(2x_1 + 3x_2 \geq 20\) where 
\[\text{Dom}(x_1) = \text{Dom}(x_2) = \{0 \ldots 8\}.
\]
- We obtain 6 clauses (instead of 18 in the original version) by applying the optimized algorithm.

\[
\begin{align*}
p(x_2 \geq 2) \\
p(x_2 \geq 3) \lor p(x_1 \geq 7) \\
p(x_2 \geq 4) \lor p(x_1 \geq 6) \\
p(x_2 \geq 5) \lor p(x_1 \geq 4) \\
p(x_2 \geq 6) \lor p(x_1 \geq 3) \\
p(x_2 \geq 7) \lor p(x_1 \geq 1)
\end{align*}
\]
Boolean Cardinality constraints (BC constraints for short) are constraints of the form $x_1 + x_2 + \cdots + x_n \# k$ where

- $k$ is an integer constant,
- $x_i$’s are literals taking a value 0 (false) or 1 (true), and
- $\#$ is one of the relations $\{\geq, \leq, =, \neq\}$.

$x_1 + x_2 + \cdots + x_n$ and $k$ are called the LHS and RHS of the constraint respectively.

Without loss of generality, we can assume the relation $\#$ is $\leq$.

\[
\begin{align*}
\sum_{i=1}^{n} x_i &\geq k \iff \sum_{i=1}^{n} \overline{x_i} \leq n - k \\
\sum_{i=1}^{n} x_i &= k \iff \sum_{i=1}^{n} x_i \leq k \land \sum_{i=1}^{n} x_i \geq k \\
\sum_{i=1}^{n} x_i &\neq k \iff \sum_{i=1}^{n} x_i \leq k - 1 \lor \sum_{i=1}^{n} x_i \geq k + 1
\end{align*}
\]
Now, we consider to apply the optimized order encoding to BC constraint $\sum_{i=1}^{n} x_i \leq k$.

**(k + 1)-wise tuple encoding**

When the order encoding is directly applied to the BC constraint, we will obtain $(k + 1)$-wise tuple encoding (pairwise when $k = 1$).

$$\neg x_{i_1} \lor \cdots \lor \neg x_{i_{k+1}} \quad (1 \leq i_1 < \cdots < i_{k+1} \leq n)$$
Order Encoding of BC (2/3)

We can derive Totalizer encoding [Bailleux and Boufkhad 2003] by introducing new integer variables for each node in the dichotomic decomposition of the summation.

**Totalizer encoding**

When we introduce new integer variables.

\[
\begin{align*}
    s_1 & \geq x_1 + x_2 \\
    s_3 & \geq x_3 + x_4 \\
    s_1 & \geq s_1 + s_3 + s_1 + s_3 \\
    \vdots \\
    s_1 & \geq s_1 + s_3 + s_1 + s_3 + \cdots \\
    s_1 & \geq s_1 + s_3 + s_1 + s_3 + \cdots + s_{n/2} + s_{n/2+1} + \cdots \\
    s_1 & \leq k
\end{align*}
\]
We can derive Sequential Counter encoding [Sinz 2005] by introducing new integer variables for each node in the sequential decomposition of the summation.

**Totalizer encoding**

When we introduce new integer variables.

\[
\begin{align*}
    s_1 & \geq x_1 \\
    s_2 & \geq x_2 + s_1 \\
    s_3 & \geq x_3 + s_2 \\
    \quad & \ldots \\
    s_n & \geq x_n + s_{n-1} \\
    s_n & \leq k
\end{align*}
\]
Pseudo-Boolean constraints (PB constraints for short) are constraints of the form \(a_1x_1 + a_2x_2 + \cdots + a_nx_n \# k\) where
- \(a_i\)'s and \(k\) are integer constants,
- \(x_i\)'s are literals taking a value 0 (false) or 1 (true), and
- \(\#\) is one of the relations \(\geq, \leq, =, \neq\).

\(a_1x_1 + a_2x_2 + \cdots + a_nx_n\) and \(k\) are called the LHS and RHS of the constraint respectively.

Without loss of generality, we can assume:
- Literals \(x_i\)'s are all distinct and no complementary pairs.
- The coefficients \(a_i\)'s are all positive and no common divisors except 1.
- The RHS value \(k\) is non-negative.
Pseudo-Boolean Satisfaction

- PB is a generalization of SAT.

\[ x_1 \lor x_2 \lor \cdots \lor x_n \iff x_1 + x_2 + \cdots + x_n \geq 1 \]

- Arithmetic constraints on integers can be represented as a PB constraint by using binary number representation.

\[ \sum_{i=0}^{m-1} 2^i x_i \]

- Disjunctions of PB constraints can be represented by a conjunction of PB constraints.

\[ \sum_{i=1}^{m} a_i x_i \geq k \lor \sum_{j=1}^{n} b_j y_j \geq l \iff p + q \geq 1 \land k \overline{p} + \sum_{i=1}^{m} a_i x_i \geq k \land l \overline{q} + \sum_{j=1}^{n} b_j y_j \geq l \]

- Satisfiability testing of one PB constraint \( \sum a_i x_i = k \) is already NP-complete (subset sum problem).
PB Solvers

PB solver is a program to solve the given PB instance. It finds an assignment satisfying all given PB constraints and sometimes also minimizing/maximizing the given objective function.

- Many good PB solvers can be found at PB Competition annually held since 2005.
  - Sat4j [Le Berre and Parrain 2010]
  - clasp [Gebser et al. 2007]
  - bsolo [Manquinho and Silva 2006]
  - wbo [Manquinho et al. 2010]
  - MiniSat+ [Één and Sörensson 2006], etc.
- Also, there are many attempts of encoding PB constraints to SAT including:
  - [Één and Sörensson 2006] (Adder+BDD+Sorter)
  - [Bailluex et al. 2006] (BDD-like)
  - [Bailluex et al. 2009] (WatchDog)
  - [Abio et al. 2012] (BDD)
Now, we consider to apply the optimized order encoding to PB constraint \( \sum_{i=1}^{n} a_i x_i \neq k \).

However, its direct application generates too many clauses, that is, we will obtain \( 2^{n-1} \) clauses in general.
Order Encoding of PB (2/3)

Introducing auxiliary variables of partial sums

We introduce integer variables $s_i$’s representing the partial sum of the first $i$ terms of the LHS, that is:

$$
s_1 = a_1 x_1, \quad s_i = a_i x_i + s_{i-1} \quad (i = 2 \ldots n).$$

Each domain $\text{Dom}(s_i)$ can be defined by $\{\sum A \mid A \subset \{a_1, \ldots, a_i\}\}$.

Example when the LHS is $3x_1 + 2x_2$

$$
s_1 = 3x_1
$$
$$
s_2 = 2x_2 + s_1
$$

$\text{Dom}(s_1) = \{0, 3\}$

$\text{Dom}(s_2) = \{0, 2, 3, 5\}$
Then, we obtain the followings by applying the order encoding.

\[
\begin{align*}
    x_1 \lor \neg p(s_1 \geq a_1) & \quad (1) \\
    \neg x_1 \lor p(s_1 \geq a_1) & \quad (2) \\
    p(s_{i-1} \geq d - a_i) \lor \neg p(s_i \geq d) & \quad (d \in \text{Dom}(s_i)) \quad (3) \\
    x_i \lor p(s_{i-1} \geq d) \lor \neg p(s_i \geq d) & \quad (d \in \text{Dom}(s_i)) \quad (4) \\
    \neg p(s_{i-1} \geq d) \lor p(s_i \geq d) & \quad (d \in \text{Dom}(s_{i-1})) \quad (5) \\
    \neg x_i \lor \neg p(s_{i-1} \geq d) \lor p(s_i \geq d + a_i) & \quad (d \in \text{Dom}(s_{i-1})) \quad (6)
\end{align*}
\]

- (1) and (2) correspond to \( s_1 \leq a_1 x_1 \) and \( s_1 \geq a_1 x_1 \) respectively.
- (3) and (4) correspond to \( s_i \leq a_1 x_1 + s_{i-1} \).
- (5) and (6) correspond to \( s_i \geq a_1 x_1 + s_{i-1} \).

The obtained encoding is equivalent to the one in [Bailleux et al. 2006] except the definitions of \( \text{Dom}(s_i) \).
Counter Matrix (1/2)

Boolean variables \( p(s_i \geq j) \) compose a sparse Boolean matrix \((s_{i,j})\) named *Counter Matrix* (CM for short).

Each component \( s_{i,j} \) satisfies:

\[
s_{i,j} \iff a_1x_1 + \cdots + a_ix_i \geq j
\]

Therefore, PB constraint \( \sum_{i=1}^{n} a_ix_i \not\equiv k \) can be represented by:

\[
\begin{align*}
  s_{n,k} & \quad \text{(when } \not\equiv \text{ is } \geq) \\
  \neg s_{n,k+1} & \quad \text{(when } \not\equiv \text{ is } \leq) \\
  s_{n,k} \land \neg s_{n,k+1} & \quad \text{(when } \not\equiv \text{ is } =) \\
  \neg s_{n,k} \lor s_{n,k+1} & \quad \text{(when } \not\equiv \text{ is } \neq)
\end{align*}
\]
Counter Matrix (2/2)

Features of Counter Matrix

- The column size of the CM can be limited to \( k + 1 \).
  - The numbers of aux. variables and clauses are \( O(nk) \).
- CM can be reused for other constraints with the same LHS.
  - We need only one CM for \( \sum_{i=1}^{n} a_i x_i \geq k \) and \( \sum_{i=1}^{n} a_i x_i \leq k \).
- It is also possible to reuse a part of CM in another PB constraint having the same sub-terms in the head.
  - \( C_1 : \ a_1 x_1 + \cdots + a_i x_i + \cdots \ # k \)
  - \( C_2 : \ a_1 x_1 + \cdots + a_i x_i + \cdots \ #' k' \)
  - We can reused the first \( i \) rows of the CM when \( k \geq k' \).

- Compared with the encoding in [Bailleux et al. 2006], the proposed encoding enables more sharings of CMs and will contribute to decrease the size of generated CNF (and hopefully improve the efficiency).
We implemented the encoding program as **PBSugar**.

- PB constraint with at most three literals is directly encoded to CNF by the order encoding.
- Otherwise, the CM is constructed for the constraint.
  - Reusing and Sharing caches contain at most 1000 CMs and is managed by LRU strategy.

- PBSugar is written in Java.
- **Glucose** version 2.1 with SatELite preprocessor is used as the default back-end SAT solver.
- For calculating the domains of $s_i$’s, balanced Diet (discrete interval encoding tree) structure proposed in [Ohnishi et al. 2003] is used.
Experimental Results

We made a comparison with the following 5 solvers.

<table>
<thead>
<tr>
<th>Solver</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sat4j PB Res//CP v2.3.2</td>
<td>[Le Berre and Parrain 2010]</td>
</tr>
<tr>
<td>clasp v2.1.1 (64-bit code)</td>
<td>[Gebser et al. 2007]</td>
</tr>
<tr>
<td>bsolo v3.2 (32-bit code)</td>
<td>[Manquinho and Silva 2006]</td>
</tr>
<tr>
<td>wbo v1.4a (32-bit code)</td>
<td>[Manquinho et al. 2010]</td>
</tr>
<tr>
<td>MiniSat+ v1.0 (64-bit code)</td>
<td>[Eén and Sörensson 2006]</td>
</tr>
</tbody>
</table>

- Sat4j PB Res//CP v2.3.2 was ranked first in 2012.
- clasp v2.1.1 is a newer version of clasp v2.0.6 which was ranked second in 2012.
- bsolo v3.2 was ranked third in 2010.
- wbo v1.4a was ranked fourth in 2010.
- MiniSat+ v1.0 was ranked third in 2006.
Experimental Results

- We used 669 instances of DEC-SMALLINT-LIN (decision problems, small integers, linear constraints) category of 2011 PB competition benchmark set.
  - classified to 22 families as done in [Abio et al. 2012]
- We compared
  - the number of solved instances within the time limit of 1000 seconds
  - on Intel Dual Core Xeon 3.33GHz machine.
## Number of solved instances

<table>
<thead>
<tr>
<th>Family</th>
<th>VBS</th>
<th>PBSugar</th>
<th>Sat4j</th>
<th>clasp</th>
<th>bsolo</th>
<th>wbo</th>
<th>MiniSat+</th>
</tr>
</thead>
<tbody>
<tr>
<td>lopes (200)</td>
<td>86</td>
<td>81</td>
<td>44</td>
<td>67</td>
<td>36</td>
<td>65</td>
<td>64</td>
</tr>
<tr>
<td>army (12)</td>
<td>11</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>6</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>blast (8)</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>cache (9)</td>
<td>9</td>
<td>9</td>
<td>6</td>
<td>9</td>
<td>6</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>chnl (21)</td>
<td>21</td>
<td>6</td>
<td>21</td>
<td>3</td>
<td>21</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>dbstv (15)</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>dlx (3)</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>elf (5)</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>fpga (36)</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>34</td>
<td>36</td>
<td>36</td>
<td>33</td>
</tr>
<tr>
<td>j30-120 (80)</td>
<td>67</td>
<td>66</td>
<td>66</td>
<td>67</td>
<td>64</td>
<td>66</td>
<td>63</td>
</tr>
<tr>
<td>neos (4)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>ooo (19)</td>
<td>18</td>
<td>15</td>
<td>15</td>
<td>17</td>
<td>12</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>pig-card (20)</td>
<td>20</td>
<td>2</td>
<td>20</td>
<td>2</td>
<td>19</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>pig-cl (20)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>ppp (6)</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>robin (6)</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>13queen (100)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>11tsp11 (100)</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>85</td>
<td>100</td>
<td>95</td>
</tr>
<tr>
<td>vdw (5)</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total (669)</td>
<td>515</td>
<td>467</td>
<td>457</td>
<td>453</td>
<td>428</td>
<td>441</td>
<td>433</td>
</tr>
</tbody>
</table>
Cactus plot

CPU time (seconds) vs. Number of solved instances for various solvers:
- PBSugar
- Sat4j
- clasp
- wbo
- MiniSat+
- bsolo

N. Tamura, M. Banbara, T. Soh (Kobe University)
We presented:

- the optimized version of the order encoding applicable to non-contiguous domain variables,
- the encoding of PB constraints by using the optimized version,
- the idea of Counter Matrix which allows reusing/sharing of clauses among different PB constraints,
- the implementation named PBSugar, and
- the experimental results for 669 DEC-SMALLINT-LIN instances.
<table>
<thead>
<tr>
<th>Outline</th>
<th>Order Encoding</th>
<th>BC Constraints</th>
<th>PB Constraints</th>
<th>PBSugar</th>
<th>Conclusion</th>
<th>Backup</th>
</tr>
</thead>
<tbody>
<tr>
<td>N. Tamura, M. Banbara, T. Soh (Kobe University)</td>
<td>PBSugar</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Outline**
- Order Encoding
- BC Constraints
- PB Constraints
- PBSugar
- Conclusion
- Backup

**PBSugar**
Order Encoding: Optimized Version

Function Encode($a_1 x_1 + \cdots + a_n x_n \geq c$)

1: if $n = 1$ then
2:   if $a_1 > 0$ then
3:     $\Psi \leftarrow \{P(x_1 \geq \lceil c/a_1 \rceil)\}$
4:   else $\{a_1 < 0\}$
5:     $\Psi \leftarrow \{-P(x_1 \geq \lceil c/a_1 \rceil + 1)\}$
6:   end if
7: else $\{n > 1\}$
8:   $D \leftarrow \{d \in \text{Dom}(x) \mid c - \sup(\sum_{i=2}^{n} a_i x_i) \leq a_1 d < c - \inf(\sum_{i=2}^{n} a_i x_i)\}$
9:   $D' \leftarrow \{d \in \text{Dom}(x) \mid a_1 d < c - \sup(\sum_{i=2}^{n} a_i x_i)\}$
10:  if $a_1 > 0$ then
11:     $\Psi \leftarrow \{P(x_1 \geq d + 1)\} \cup \theta \mid d \in D, \theta \in \text{Encode}(a_2 x_2 + \cdots + a_n x_n \geq c - a_1 d)$
12:     if $D' \neq \emptyset$ then
13:        $\Psi \leftarrow \{P(x_1 \geq \max(D') + 1)\} \cup \Psi$
14:     end if
15:  else $\{a_1 < 0\}$
16:     $\Psi \leftarrow \{-P(x_1 \geq d)\} \cup \theta \mid d \in D, \theta \in \text{Encode}(a_2 x_2 + \cdots + a_n x_n \geq c - a_1 d)$
17:     if $D' \neq \emptyset$ then
18:        $\Psi \leftarrow \{-P(x_1 \geq \min(D'))\} \cup \Psi$
19:     end if
20:  end if
21: end if
22: return $\Psi$
### Number of auxiliary variables (in thousands)

<table>
<thead>
<tr>
<th>Family</th>
<th>PBSugar</th>
<th>Adder</th>
<th>WD1</th>
<th>WD2</th>
<th>BDD1</th>
<th>BDD2</th>
<th>BDD3</th>
</tr>
</thead>
<tbody>
<tr>
<td>lopes (200)</td>
<td>54.07</td>
<td>1744.05</td>
<td>3566.11</td>
<td>5478.81</td>
<td>2393.97</td>
<td>2394.00</td>
<td>7734.65</td>
</tr>
<tr>
<td>army (12)</td>
<td>4.66</td>
<td>4.63</td>
<td>10.96</td>
<td>245.49</td>
<td>6.36</td>
<td>6.36</td>
<td>479.83</td>
</tr>
<tr>
<td>blast (8)</td>
<td>2738.13</td>
<td>27.77</td>
<td>62.22</td>
<td>1394.50</td>
<td>36.73</td>
<td>39.67</td>
<td>761.29</td>
</tr>
<tr>
<td>cache (9)</td>
<td>729.06</td>
<td>145.66</td>
<td>339.18</td>
<td>2393.97</td>
<td>201.18</td>
<td>210.83</td>
<td>1503.19</td>
</tr>
<tr>
<td>chnl (21)</td>
<td>4.52</td>
<td>8.39</td>
<td>24.55</td>
<td>1007.90</td>
<td>6.76</td>
<td>6.76</td>
<td>184.59</td>
</tr>
<tr>
<td>dbstv30 (5)</td>
<td>39.99</td>
<td>219.82</td>
<td>709.73</td>
<td>–</td>
<td>441.86</td>
<td>1695.87</td>
<td>–</td>
</tr>
<tr>
<td>dbstv40 (5)</td>
<td>90.98</td>
<td>2468.45</td>
<td>6564.44</td>
<td>–</td>
<td>4282.16</td>
<td>7225.63</td>
<td>–</td>
</tr>
<tr>
<td>dbstv50 (5)</td>
<td>187.40</td>
<td>6135.13</td>
<td>16365.39</td>
<td>–</td>
<td>11111.06</td>
<td>19723.37</td>
<td>–</td>
</tr>
<tr>
<td>dlx (3)</td>
<td>41.53</td>
<td>10.40</td>
<td>21.62</td>
<td>247.79</td>
<td>12.40</td>
<td>13.89</td>
<td>126.81</td>
</tr>
<tr>
<td>elf (5)</td>
<td>103.15</td>
<td>20.37</td>
<td>42.78</td>
<td>571.38</td>
<td>24.62</td>
<td>28.13</td>
<td>306.76</td>
</tr>
<tr>
<td>fpga (36)</td>
<td>1.82</td>
<td>21.15</td>
<td>53.96</td>
<td>1074.03</td>
<td>13.27</td>
<td>13.27</td>
<td>242.03</td>
</tr>
<tr>
<td>j120 (28)</td>
<td>698.46</td>
<td>159.43</td>
<td>540.00</td>
<td>5713.75</td>
<td>612.13</td>
<td>806.08</td>
<td>2246.82</td>
</tr>
<tr>
<td>j30 (17)</td>
<td>70.16</td>
<td>18.15</td>
<td>50.80</td>
<td>1190.59</td>
<td>44.96</td>
<td>59.82</td>
<td>1153.51</td>
</tr>
<tr>
<td>j60 (18)</td>
<td>218.58</td>
<td>37.03</td>
<td>112.35</td>
<td>4775.72</td>
<td>157.92</td>
<td>180.05</td>
<td>7285.69</td>
</tr>
<tr>
<td>j90 (17)</td>
<td>770.31</td>
<td>65.41</td>
<td>217.01</td>
<td>6543.52</td>
<td>553.80</td>
<td>553.76</td>
<td>19793.50</td>
</tr>
<tr>
<td>neos (4)</td>
<td>62.69</td>
<td>73.74</td>
<td>185.59</td>
<td>3542.94</td>
<td>79.33</td>
<td>122.53</td>
<td>2003.95</td>
</tr>
<tr>
<td>ooo (19)</td>
<td>818.80</td>
<td>118.15</td>
<td>273.54</td>
<td>2248.35</td>
<td>162.25</td>
<td>168.61</td>
<td>1315.77</td>
</tr>
<tr>
<td>pig-card (20)</td>
<td>7.97</td>
<td>15.26</td>
<td>50.75</td>
<td>2966.58</td>
<td>11.94</td>
<td>11.94</td>
<td>632.33</td>
</tr>
<tr>
<td>pig-cl (20)</td>
<td>0.00</td>
<td>7.68</td>
<td>25.25</td>
<td>1984.06</td>
<td>4.01</td>
<td>4.01</td>
<td>310.03</td>
</tr>
<tr>
<td>ppp (6)</td>
<td>26.00</td>
<td>57.13</td>
<td>141.49</td>
<td>623.59</td>
<td>81.57</td>
<td>82.86</td>
<td>382.68</td>
</tr>
<tr>
<td>robin (6)</td>
<td>123.37</td>
<td>171.67</td>
<td>628.45</td>
<td>3634.13</td>
<td>158.55</td>
<td>158.55</td>
<td>16565.75</td>
</tr>
<tr>
<td>13queen (100)</td>
<td>5.87</td>
<td>2.20</td>
<td>6.18</td>
<td>461.54</td>
<td>5.63</td>
<td>7.08</td>
<td>791.43</td>
</tr>
<tr>
<td>11tsp11 (100)</td>
<td>2.35</td>
<td>3.37</td>
<td>8.83</td>
<td>170.59</td>
<td>5.71</td>
<td>6.51</td>
<td>221.21</td>
</tr>
<tr>
<td>vdw (5)</td>
<td>2.23</td>
<td>1895.39</td>
<td>3356.94</td>
<td>12818.51</td>
<td>1391.65</td>
<td>1391.65</td>
<td>5875.58</td>
</tr>
<tr>
<td>Average (669)</td>
<td>145.44</td>
<td>602.15</td>
<td>1289.35</td>
<td>2324.57</td>
<td>896.64</td>
<td>1002.94</td>
<td>4068.87</td>
</tr>
</tbody>
</table>

Comparison with the results reported in [Abio et al. 2012]
## Number of clauses (in thousands)

<table>
<thead>
<tr>
<th>Family</th>
<th>PBSugar</th>
<th>Adder</th>
<th>WD1</th>
<th>WD2</th>
<th>BDD1</th>
<th>BDD2</th>
<th>BDD3</th>
</tr>
</thead>
<tbody>
<tr>
<td>lopes (200)</td>
<td>1397.73</td>
<td>10901.95</td>
<td>7730.20</td>
<td>22171.19</td>
<td>3436.72</td>
<td>3436.67</td>
<td>10004.63</td>
</tr>
<tr>
<td>army (12)</td>
<td>24.03</td>
<td>26.29</td>
<td>34.69</td>
<td>2155.89</td>
<td>11.07</td>
<td>11.06</td>
<td>925.12</td>
</tr>
<tr>
<td>blast (8)</td>
<td>12316.27</td>
<td>191.46</td>
<td>108.58</td>
<td>2114.99</td>
<td>77.41</td>
<td>71.97</td>
<td>1271.34</td>
</tr>
<tr>
<td>cache (9)</td>
<td>3437.40</td>
<td>1161.61</td>
<td>731.78</td>
<td>3833.01</td>
<td>453.37</td>
<td>456.36</td>
<td>2599.87</td>
</tr>
<tr>
<td>chnl (21)</td>
<td>20.22</td>
<td>56.82</td>
<td>116.97</td>
<td>4936.16</td>
<td>11.23</td>
<td>11.23</td>
<td>285.67</td>
</tr>
<tr>
<td>dbstv30 (5)</td>
<td>522.96</td>
<td>1823.61</td>
<td>3693.95</td>
<td>–</td>
<td>1183.59</td>
<td>3608.42</td>
<td>–</td>
</tr>
<tr>
<td>dbstv40 (5)</td>
<td>1434.74</td>
<td>18169.82</td>
<td>17901.80</td>
<td>–</td>
<td>6511.80</td>
<td>12244.49</td>
<td>–</td>
</tr>
<tr>
<td>dbstv50 (5)</td>
<td>3480.32</td>
<td>45349.38</td>
<td>46862.83</td>
<td>–</td>
<td>16952.14</td>
<td>33831.20</td>
<td>–</td>
</tr>
<tr>
<td>dlx (3)</td>
<td>204.18</td>
<td>86.15</td>
<td>56.59</td>
<td>398.67</td>
<td>43.94</td>
<td>43.49</td>
<td>229.83</td>
</tr>
<tr>
<td>elf (5)</td>
<td>505.38</td>
<td>175.49</td>
<td>117.73</td>
<td>927.47</td>
<td>92.75</td>
<td>92.51</td>
<td>553.90</td>
</tr>
<tr>
<td>fpga (36)</td>
<td>8.69</td>
<td>139.45</td>
<td>175.66</td>
<td>3615.21</td>
<td>15.65</td>
<td>15.65</td>
<td>278.37</td>
</tr>
<tr>
<td>j120 (28)</td>
<td>3530.43</td>
<td>1207.00</td>
<td>3907.42</td>
<td>26283.89</td>
<td>1290.70</td>
<td>1674.87</td>
<td>44157.85</td>
</tr>
<tr>
<td>j30 (17)</td>
<td>348.67</td>
<td>135.25</td>
<td>178.47</td>
<td>3903.27</td>
<td>103.22</td>
<td>129.73</td>
<td>2257.11</td>
</tr>
<tr>
<td>j60 (18)</td>
<td>1097.27</td>
<td>283.99</td>
<td>525.65</td>
<td>22873.88</td>
<td>341.98</td>
<td>381.84</td>
<td>14386.03</td>
</tr>
<tr>
<td>j90 (17)</td>
<td>3851.46</td>
<td>501.05</td>
<td>1336.72</td>
<td>34184.53</td>
<td>1156.78</td>
<td>1145.67</td>
<td>39163.47</td>
</tr>
<tr>
<td>neos (4)</td>
<td>288.47</td>
<td>472.92</td>
<td>596.17</td>
<td>12411.55</td>
<td>140.71</td>
<td>221.90</td>
<td>3682.71</td>
</tr>
<tr>
<td>ooo (19)</td>
<td>3751.32</td>
<td>888.58</td>
<td>537.69</td>
<td>3473.38</td>
<td>314.62</td>
<td>322.82</td>
<td>2221.64</td>
</tr>
<tr>
<td>pig-card (20)</td>
<td>35.75</td>
<td>104.68</td>
<td>367.03</td>
<td>20711.11</td>
<td>19.86</td>
<td>19.86</td>
<td>958.65</td>
</tr>
<tr>
<td>pig-cl (20)</td>
<td>161.21</td>
<td>213.56</td>
<td>341.48</td>
<td>14802.42</td>
<td>165.22</td>
<td>165.22</td>
<td>475.20</td>
</tr>
<tr>
<td>ppl (6)</td>
<td>147.10</td>
<td>422.45</td>
<td>301.51</td>
<td>1834.14</td>
<td>130.73</td>
<td>133.08</td>
<td>684.32</td>
</tr>
<tr>
<td>robin (6)</td>
<td>562.68</td>
<td>1185.92</td>
<td>6916.35</td>
<td>19694.86</td>
<td>281.42</td>
<td>281.42</td>
<td>28875.61</td>
</tr>
<tr>
<td>13queen (100)</td>
<td>28.31</td>
<td>14.74</td>
<td>38.92</td>
<td>5068.36</td>
<td>10.85</td>
<td>13.74</td>
<td>1573.90</td>
</tr>
<tr>
<td>11tsp11 (100)</td>
<td>13.98</td>
<td>25.97</td>
<td>28.01</td>
<td>1338.27</td>
<td>10.43</td>
<td>12.01</td>
<td>436.26</td>
</tr>
<tr>
<td>vdw (5)</td>
<td>287.18</td>
<td>10894.94</td>
<td>6573.12</td>
<td>24283.19</td>
<td>1671.71</td>
<td>1671.71</td>
<td>7271.86</td>
</tr>
<tr>
<td>Average (669)</td>
<td>1070.76</td>
<td>3864.52</td>
<td>3146.49</td>
<td>10316.25</td>
<td>1341.46</td>
<td>1548.42</td>
<td>6339.21</td>
</tr>
</tbody>
</table>

- Comparison with the results reported in [Abio et al. 2012]