A Pigeon-Hole Based Encoding of Cardinality Constraints

Said Jabbour and Lakhdar Sais and Yakoub Salhi
CRIL - CNRS, Université d'Artois, France
{jabbour, sais, salhi}@cril.fr

Abstract

In this paper, we propose a new encoding of the cardinality constraint $\sum_{i=1}^{n} x_i \geqslant b$. It makes an original use of the general formulation of the Pigeon-Hole principle to derive a formula in conjunctive normal form (CNF). Our Pigeon-Hole based CNF encoding can be seen as a simple way to express the semantic of the cardinality constraint, that can be defined as how to put b pigeons into n holes. To derive an efficient CNF encoding that ensures constraint propagation, we exploit the set of symmetries of the Pigeon-Hole based formulation to derive an efficient CNF encoding of the cardinality constraint. More interestingly, the final CNF formula contains is $b \times (n-b)$ variables and clauses and belongs to the well-known Reverse-Horn tractable CNF formula, which can be decided by unit propagation. Our proposed Pigeon-Hole based encoding is theoretically compared with the currently well-known CNF encoding of the cardinality constraint.

Introduction

Today, Boolean satisfiability (SAT) has gained a considerable audience with the advent of a new generation of solvers able to solve large instances encoding real-world problems. In addition to the traditional applications of SAT to hardware and software formal verification, this impressive progress led to increasing use of SAT technology to solve new real-world applications such as planning, bioinformatics, cryptography, and data mining. Encoding applications as formulas in CNF became now a usual practice.

One of the most important flaws of CNF or Boolean representation in general rises in the difficulty to deal with counting constraints, among them the cardinality constraint and its more general form the pseudo Boolean constraint. Indeed, several applications involve counting arguments expressed as cardinality or pseudo Boolean constraint. This kind of constraints arises frequently out of the encoding of real-world problems such as radio frequency assignment, time tabling and product configuration problems to cite a few. For the above reasons, several authors have addressed the issue of finding an efficient encoding of cardinality (e.g. (Warners 1996; Bailleux and Boufkhad 2003; Sinz 2005; Silva and Lynce 2007; Asín et al. 2009)) and pseudo Boolean constraints (e.g. (Eén and Sörensson 2006; Bailleux, Boufkhad, and Roussel 2009)) as a CNF formula. Efficiency refers to both the compactness of the representation (size of the CNF formula) and to the ability to achieve the same level of constraint propagation (generalized arc consistency) on the CNF formula.

In this paper, we present a new encoding of the cardinality constraint as a formula in CNF. Our proposed encoding starts from an initial formulation of this counting constraint using the well known Pigeon-Hole principle. This first CNF encoding is not effective as one must face satisfiability checking of a hard CNF sub-formula expressing the well-known Pigeon-Hole problem. To avoid this problem, we combine several techniques to derive the final and efficient CNF encoding of the cardinality constraint. More precisely, our method is based on the application of the resolution rule using different kinds of clauses: symmetry breaking predicates, blocked clauses and redundant clauses. The resulting CNF formula is very natural, and preserves generalized arc consistency through unit propagation.

Technical Background and Preliminary Definitions

Preliminary Definitions and Notations

A Boolean formula $\mathcal{F}$ in *Conjunctive Normal Form (CNF)* is a conjunction of *clauses*, where a clause is a disjunction of *literals*. A literal is a positive (x) or negated ($\neg x$) propositional variable. The two literals x and $\neg x$ are called *complementary*. We denote by $\tilde{l}$ the complementary literal of l. More precisely, if $l = x$ then $\tilde{l} = \neg x$, otherwise $\tilde{l} = x$. The variable associated to a literal l is denoted by $|l|$. Let us recall that any Boolean formula can be translated to CNF using linear Tseitin's encoding (Tseitin 1968). A *unit* clause is a clause containing only one literal (called *unit literal*), while a binary clause contains exactly two literals. A Horn (resp. reverse Horn) clause is a clause with at-most one positive (resp. negative) literal. A positive (resp. negative) clause is a clause whose literals are all positive (resp. negative). An *empty clause*, denoted $\bot$, is interpreted as false (unsatisfiable), whereas an *empty CNF formula*, denoted $\top$, is interpreted as true (satisfiable).

The set of variables occurring in $\mathcal{F}$ is denoted $V_{\mathcal{F}}$ and its associated set of literals $\mathcal{L}_{\mathcal{F}} = \cup_{x \in \mathcal{V}_{\mathcal{F}}} \{x, \neg x\}$. A set of literals is *complete* if it contains one literal for each variable in $V_{\mathcal{F}}$, and *fundamental* if it does not contain complementary literals. A literal l is called *monotone or pure* if $\tilde{l}$ does not appear in $\mathcal{F}$. An *interpretation* ρ of a Boolean formula $\mathcal{F}$ is

a function which associates a truth value $\rho(x) \in \{0,1\}$ (0 for false and 1 for $true$) to some of the variables $x \in V_{\mathcal{F}}$. ρ is *complete* if it assigns a value to every $x \in V_{\mathcal{F}}$, and *partial* otherwise. An interpretation is alternatively represented by a complete and fundamental set of literals. A *model* of a formula $\mathcal{F}$ is an interpretation ρ that satisfies the formula, denoted $\rho \models \mathcal{F}$. A formula $\mathcal{G}$ is a logical consequence of a formula $\mathcal{F}$, denoted $\mathcal{F} \models \mathcal{G}$, iff every model of $\mathcal{F}$ is a model of $\mathcal{G}$.

Let c_i and c_j be two clauses such that $c_i = (x \vee \alpha)$ and $c_j = (\neg x \vee \beta)$, $\eta[x, c_i, c_j] = (\alpha \vee \beta)$ denotes the *resolvent* on x between c_i and c_j. A resolvent is called *tautological* when it contains complementary literals.

$\mathcal{F}|_x$ denotes the formula obtained from $\mathcal{F}$ by assigning x the truth-value $true$. Formally $\mathcal{F}|_x = \{c \mid c \in \mathcal{F}, \{x, \neg x\} \cap c = \emptyset\} \cup \{c \backslash \{\neg x\} \mid c \in \mathcal{F}, \neg x \in c\}$. This notation is extended to interpretations: given an interpretation $\rho = \{x_1, \ldots, x_n\}$, we define $\mathcal{F}|_\rho = (\ldots((\mathcal{F}|_{x_1})|_{x_2})\ldots|_{x_n})$.

$\mathcal{F}^*$ denotes the formula $\mathcal{F}$ closed under unit propagation, defined recursively as follows: (1) $\mathcal{F}^* = \mathcal{F}$ if $\mathcal{F}$ does not contain any unit clause, (2) $\mathcal{F}^* = \perp$ if $\mathcal{F}$ contains two unit-clauses $\{x\}$ and $\{\neg x\}$, (3) otherwise, $\mathcal{F}^* = (\mathcal{F}|_x)^*$ where x is the literal appearing in a unit clause of $\mathcal{F}$.

Let c_1 and c_2 be two clauses of a formula $\mathcal{F}$. We say that c_1 (respectively c_2) subsume (resp. is subsumed) c_2 (resp. by c_1) iff $c_1 \subseteq c_2$. If c_1 subsume c_2, then $c_1 \models c_2$ (the converse is not true).

Let $c \in \mathcal{F}$ such that $x \in c$, the literal x of c is called *blocked* if $\forall c' \in \mathcal{F}$ such that $\neg x \in c'$ and $c \neq c'$, $\eta[x, c, c']$ is a tautology. A clause $c \in \mathcal{F}$ is a *blocked clause* if it contains a blocked literal (Kullmann 1997). A blocked clause $c \in \mathcal{F}$ can be deleted from $\mathcal{F}$ while preserving satisfiability.

Pigeon-Hole Principle

Our encoding is based on the Pigeon-Hole principle widely used in proof complexity. It asserts that there is no injective mapping from b pigeons to n holes as long as $b > n$. Stephen A. Cook proved that the propositional formula encoding the Pigeon-Hole problem have polynomial size proof in extended resolution proof system (Cook 1976). A polynomial proof is also obtained by Krishnamurthy (Krishnamurthy 1985) using resolution with symmetry. The Pigeon-Hole principle PHP_n^b can be expressed as a propositional formula in conjunctive normal form. The variables of PHP_n^b are p_{ij} with $1 \leqslant i \leqslant b$, $1 \leqslant j \leqslant n$; the variable p_{ij} is intended to denote the condition that pigeon i is sitting in hole j. The CNF formula encoding PHP_n^b can be stated as follows:

$$\bigvee_{j=1}^{n} p_{ij}, \quad 1 \leqslant i \leqslant b \tag{1}$$

$$\bigwedge_{1 \leqslant i < k \leqslant b} (\neg p_{ij} \vee \neg p_{kj}), \quad 1 \leqslant j \leqslant n \tag{2}$$

The first equation (1) expresses that any pigeon must be put in at least one hole, while the equation (2) constrains each hole to contain at most one pigeon.

Symmetries in SAT

As our Pigeon-Hole based encoding heavily exploit symmetries (Krishnamurthy 1985), we briefly recall the symmetry breaking framework in SAT. For more details on symmetry, we refer the reader to some but not exhaustive list of works in SAT (Benhamou and Sais 1992; 1994; Crawford et al. 1996; Aloul et al. 2003) and CSP (Puget 1993; Gent, Petrie, and Puget 2006).

First, let us introduce some definitions on group theory. A group $(\mathcal{G}, \circ)$ is a finite set $\mathcal{G}$ with an associative binary operation $\circ : \mathcal{G} \times \mathcal{G} \rightarrow \mathcal{G}$ admitting a neutral and an inverse element. The set of all permutation $\mathcal{P}$ over a finite set E associated to the composition operator $\circ$, denoted $(\mathcal{P}, \circ)$, forms a group. Furthermore, each permutation $\sigma \in \mathcal{P}$ can be represented by a set of cycles $\{c_1 \ldots c_n\}$ where each cycle c_i is a list of elements of E $(l_{i_1} \ldots l_{i_{n_i}})$ s.t. $\forall 1 \leqslant k < n_i, \sigma(l_{i_k}) = l_{i_{k+1}}$ and $\sigma(l_{i_{n_i}}) = l_{i_1}$.

Let $\mathcal{F}$ be a CNF formula, and σ a permutation over $\mathcal{L}(\mathcal{F})$. We can extend the definition of the permutation σ to $\mathcal{F}$ as follows: $\sigma(\mathcal{F}) = \{\sigma(c) | c \in \mathcal{F}\}$ and $\sigma(c) = \{\sigma(l) | l \in c\}$.

Definition 1. *Let $\mathcal{F}$ be a CNF formula and σ a permutation over the literals of $\mathcal{F}$, σ is a symmetry of $\mathcal{F}$ if it satisfies the following conditions:*

- $\sigma(\neg x) = \neg\sigma(x)$, $\forall x \in \mathcal{L}_{\mathcal{F}}$
- $\sigma(\mathcal{F}) = \mathcal{F}$

From the definition above, a symmetry σ defines an equivalence relation over the set of possible assignments. We need to consider only one assignment from each equivalence class. Breaking symmetries consist in eliminating all symmetric assignments except one in each equivalence class. The most used approach to break symmetries consists in adding new clauses - called symmetry breaking predicates (SBP) or lex leader constraints - to the original formula (Crawford 1992; Crawford et al. 1996; Aloul et al. 2003; Walsh 2006).

Before introducing the general definition of SBP, let us illustrate the main idea behind this technique using a simple example. Let $\sigma = (x_1, y_1)$ be a symmetry of a CNF formula $\mathcal{F}$ with only one cycle. Suppose that $\mathcal{F}$ admits $m = \{x_1, \neg y_1 \ldots\}$ as a model, then $\sigma(m) = \{\neg x_1, y_1, \ldots\}$ is also a model of $\mathcal{F}$. To break this symmetry, it is sufficient to lay down an ordering on the values of x_1 and y_1. For example, adding conjunctively the constraint $x_1 \leqslant y_1$, which can be expressed by the clause $c = (\neg x_1 \vee y_1)$, to the formula $\mathcal{F}$, leads to a new formula $\Phi = \mathcal{F} \cup \{c\}$ while preserving satisfiability. The model m of $\mathcal{F}$ is eliminated as it is not a model of Φ. All other models of $\mathcal{F}$ not satisfying the added binary clause are also eliminated. This idea is generalized in definition 3 to a symmetry containing arbitrary number of cycles.

Definition 2. *Let $\sigma = (x_1, y_1), \ldots, (x_n, y_n)$ be a symmetry of $\mathcal{F}$. σ is called lexicographically ordered iff $\forall i (1 \leqslant i \leqslant n-1)\ |x_i| < |x_{i+1}|$ and $\forall i (1 \leqslant i \leqslant n)\ |x_i| < |y_i|$ holds.*

Definition 3 (SBP (Crawford et al. 1996))**.** *Let $\mathcal{F}$ be a CNF and $\sigma = (x_1, y_1)(x_2, y_2)\ldots(x_n, y_n)$ a symmetry of $\mathcal{F}$.*

Then the symmetry breaking predicates, called sbp_σ, associated to a lexicographically ordered symmetry σ is defined as the conjunction of the following constraints:

- $(x_1 \leqslant y_1) \wedge$
- $(x_1 = y_1) \rightarrow (x_2 \leqslant y_2) \wedge$
- $\ldots \wedge$
- $(x_1 = y_1) \wedge (x_2 = y_2) \ldots (x_{n-1} = y_{n-1}) \rightarrow (x_n \leqslant y_n)$

Similarly, in order to break a set of symmetries one need to add conjunctively symmetry breaking predicates associated to each individual symmetry.

The following property shows that symmetry breaking predicates approach preserves the satisfiability between the original formula and the generated one.

Proposition 1 ((Crawford et al. 1996)). *Let $\mathcal{F}$ be a CNF formula and σ a symmetry of $\mathcal{F}$. Then $\mathcal{F}$ and $(\mathcal{F} \wedge sbp_\sigma)$ are equivalent w.r.t. satisfiability.*

In order to limit the combinatorial explosion of the clausal transformation of these predicates, one has to add one variable α_i per cycle (x_i, y_i) to express the equality between x_i and y_i. However, one of the major drawbacks of this approach is that the size of the symmetry breaking predicates is exponential in the worst case. Recently, interesting reductions in the size of the SBP has been obtained in (Aloul, Sakallah, and Markov 2006) using non redundant generators concept.

Piegon-Hole based Encoding of cardinality Constraints

$\sum_{i=1}^{n} x_i \geqslant b$ such that x_i is propositional variable ($x_i \in \{0, 1\}$), for $1 \leqslant i \leqslant n$, is a well known cardinality constraint. As mentioned by Joot P. Warners in (Warners 1996), this kind of constraints and its generalized form $\sum_{i=1}^{n} a_i x_i \geqslant b$ (where a_i are positive integers) can be polynomially encoded as a propositional formula in CNF. The first polynomial CNF expansion of cardinality constraint is proposed by Hooker in an unpublished note (see also (Warners 1996)). Let us give the formulation of the Hooker CNF encoding of the $\sum_{i=1}^{n} x_i \geqslant b$ constraint as it is was described in (Warners 1996) (page 12):

$$(\neg z_{ik} \vee x_i), \quad 1 \leqslant i \leqslant n, \quad 1 \leqslant k \leqslant b \tag{3}$$

$$\bigvee_{i=1}^{n} z_{ik}, \quad 1 \leqslant k \leqslant b \tag{4}$$

$$(\neg z_{ik} \vee \neg z_{jk}), \quad 1 \leqslant i < j \leqslant n, \quad 1 \leqslant k \leqslant b \tag{5}$$

Let us note that in (Warners 1996) the author mentions that the formula (3) says that x_i is true if some z_{ik} is true, while formula (4) combined with formula (5) say that for each k exactly one z_{ik} must be true.

However this formulation is clearly wrong. Let us give a counter example. Suppose that $x_i = 0$ for $1 \leqslant i \leqslant n - (b - 1)$. In such a case, the cardinality constraint $\sum_{i=1}^{n} x_i \geqslant b$ is unsatisfiable as one needs to set b variables to $true$ among the set of remaining unassigned variables $R = \{x_{n-(b-2)}, x_{n-(b-3)}, \ldots, x_n\}$. Indeed, this is clearly impossible as the number of unassigned variables is $n - (n - (b-2)) + 1 = b - 1$. On the contrary, the CNF formula made of (3), (4) and (5) is satisfiable. One can set the remaining variables of R to $true$ and for each k ($1 \leqslant k \leqslant b$) set exactly one z_{ik} to $true$ for $(n - (b-2) \leqslant i \leqslant n)$.

Despite of the importance of the Warners' paper and its precursory nature on the subject, to our knowledge, this error in the formulation of the first translation of the cardinality constraint to CNF reported by Warners was never raised.

Based on the description above, the first contribution of this paper concerns the correct reformulation of the CNF representation of the cardinality constraint $\sum_{j=1}^{n} x_j \geqslant b$, that we denote by $\mathcal{P}_n^b$ in the sequel:

$$\bigwedge_{k=1}^{b} (\neg p_{ki} \vee x_i), \quad 1 \leqslant i \leqslant n \tag{6}$$

$$\bigvee_{i=1}^{n} p_{ki}, \quad 1 \leqslant k \leqslant b \tag{7}$$

$$\bigwedge_{1 \leqslant k < k' \leqslant b} (\neg p_{ki} \vee \neg p_{k'i}), \quad 1 \leqslant i \leqslant n \tag{8}$$

Let us mention that the two equations (7) and (8) encode the well-known pigeon hole problem PHP_n^b, where b is the number of pigeons and n is the number of holes (p_{ki} expresses that pigeon k is in hole i). The mapping between the models of PHP_n^b and those of $\sum_{i=1}^{n} x_i \geqslant b$ are obtained thanks to the equation (6). Indeed, the propositional variable x_i is true if the hole i contains one of the pigeons k for $1 \leqslant k \leqslant b$. If we take again the previous counter example, the CNF formula $\mathcal{P}_n^b$ becomes unsatisfiable as it encodes an unsatisfiable Pigeon-Hole problem PHP_{b-1}^b.

In this original polynomial transformation, the number of variables is $n + b \times n$ and the number of clauses required is $n \times b + b + n \times \frac{b \times (b-1)}{2}$. The overall complexity is in $\mathcal{O}(b \times n)$ variables and $\mathcal{O}(n \times b^2)$ clauses.

Unfortunately, checking the satisfiability of a Pigeon-Hole formula is computationally hard except if we use resolution with symmetry or extended resolution proof systems. In the following, we show how to improve the efficiency of this Pigeon-Hole based encoding of the cardinality constraint. By efficiency, we mean enhancing the propagation capabilities (unit propagation) of the obtained CNF. To this end, we show in the next section, how symmetries of the this Pigeon-Hole formulation can be used to enhance this first version of our encoding.

Symmetry Breaking on the Pigeon-Hole Based Encoding

In this section, we show how symmetry breaking predicates can be used to reduce the size of our Pigeon-Hole based encoding of the cardinality constraint while ensuring unit propagation.

For clarity reason, and to better visualize the reductions on our previous encoding $\mathcal{P}_n^b$, we use the following matrix representation for the CNF formula (7). Each row represents

a positive clause of (7).

$$\begin{pmatrix} p_{11} & \cdots & & & [p_{1b} & \cdots & p_{1n}] \\ p_{21} & \cdots & & [p_{2(b-1)} & \cdots & p_{2(n-1)}] & p_{2n} \\ \vdots & & \iddots & & & \iddots & \vdots \\ [p_{b1} & \cdots & p_{b(n-(b-1))}] & & & \cdots & p_{bn} \end{pmatrix}$$

Efficient Encoding We give now our enhanced CNF Pigeon-Hole based encoding, called $ph\mathcal{P}_n^b$, of a cardinality constraint:

$$\neg p_{(b-k+1)(i+k-1)} \vee x_{(i+k-1)}, \quad 1 \leqslant i \leqslant n-b+1, \; 1 \leqslant k \leqslant b \tag{9}$$

$$\bigvee_{i=1}^{n-b+1} p_{(b-k+1)(i+k-1)}, \quad 1 \leqslant k \leqslant b \tag{10}$$

$$p_{(b-k+1)k} \vee \cdots \vee p_{(b-k+1)(i+k)} \vee \neg p_{(b-k)(i+k+1)}, \; 0 \leqslant i \leqslant n-b-1, 1 \leqslant k \leqslant b-1 \tag{11}$$

This efficient $ph\mathcal{P}_n^b$ encoding is obtained from $\mathcal{P}_n^b$ encoding using sophisticated reductions. Before illustrating how such reductions are performed, let us describe briefly this encoding. The formula (10) corresponds to the reduction of (7) to only the sub-clauses represented in brackets (see the previous matrix). These sub-clauses are obtained by deducing that the literals belonging to the upper-left corner triangle and to the lower-right corner triangle of the previous matrix must be assigned to false. For instance, the clause $p_{b1} \vee \cdots \vee p_{b(n-(b-1))} \in (11)$ is obtained for $k = 1$, corresponding to the last clause in brackets of the previous matrix. Moreover, the formula (9) corresponds to the restriction of (6) to the variables appearing in (10). Finally, the formula (11), called stair-implications, link successive rows in the matrix from the bottom to the top. With these implications the set of negative binary clauses (8) are made redundant and then can be dropped. One can see that the number of clauses of (11) is smaller than that of (8).

From $ph\mathcal{P}_n^b$, one can deduce that the overall complexity of our encoding is in $\mathcal{O}(b \times (n-b))$ variables and clauses.

We now illustrate how symmetry breaking predicates can be used to improve $\mathcal{P}_n^b$ in order to obtain our final $ph\mathcal{P}_n^b$ encoding made only of the formulas (9), (10) and (11). Let us take a closer look to the CNF formula $\mathcal{P}_n^b$. Let $\mathcal{S}ym(\mathcal{P}_n^b)$ be the following set of symmetries (row symmetries) of $\mathcal{P}_n^b$:

$$\bigcup_{1 \leqslant i < j \leqslant b} \sigma(i,j) = (p_{i1}, p_{j1}), (p_{i2}, p_{j2}), \ldots, (p_{in}, p_{jn}) \tag{12}$$

The other symmetries (column symmetries) that involve the variables x_i are not considered, as these variables can take part in other constraints. In this case a permutation between the variables x_i is clearly a local or conditional symmetry (Gent et al. 2005).

We here show how the formula (7) can be reduced to only the sub-clauses represented in brackets corresponding to the formula (10). For this purpose, we only need to show that the literals belonging to the upper-left and lower-right corner triangles can be assigned to $false$. The assignment of such subset of variables allows as to derive the formula (9) from the formula (6). This reduction is obtained by applying the resolution rule between the clauses of the symmetry breaking predicates $sbp(\mathcal{S}ym(\mathcal{P}_n^b))$ and those of (7) and (8).

Eliminating The Upper-Left Corner Triangle Let us iteratively show how literals belonging to the upper-left corner triangle can be eliminated.

- $\mathcal{P}_n^b \wedge sbp(\mathcal{S}ym(\mathcal{P}_n^b)) \models \neg p_{11} \wedge \neg p_{21}, \ldots, \neg p_{(b-1)1}$ i.e. first column of the upper-left corner triangle.
  1. $\mathcal{P}_n^b \wedge sbp_{\sigma(1,2)} \models \neg p_{11}$: Let $\sigma = (p_{11}, p_{21}), (p_{12}, p_{22}) \subset \sigma(1,2)$. $sbp_\sigma = (p_{11} \leqslant p_{21}) \wedge (p_{11} = p_{21}) \rightarrow (p_{12} \leqslant p_{22})$. The CNF formula associated to $sbp(\sigma)$ is $(1)(\neg p_{11} \vee p_{21})$, $(2)(p_{11} \vee p_{21} \vee \neg p_{12} \vee p_{22})$, $(3)(\neg p_{11} \vee \neg p_{21} \vee \neg p_{12} \vee p_{22})$. The clause (3) is subsumed by the binary clause $c = (\neg p_{11} \vee \neg p_{21}) \in \mathcal{P}_n^b$ (see formula (8)). $\eta[p_{21}, (1), c] = \neg p_{11}$. By propagating $\neg p_{11}$, we eliminate p_{11} from the first row of the matrix and we satisfy all the negative binary clauses of (8) involving $\neg p_{11}$.
  2. Using the same reasoning, we can prove that $\mathcal{P}_n^b \wedge sbp(\mathcal{S}ym(\mathcal{P}_n^b)) \models \neg p_{21} \wedge \neg p_{31} \wedge, \ldots, \neg p_{(b-1)1}$.
- $\mathcal{P}_n^b \wedge sbp(\mathcal{S}ym(\mathcal{P}_n^b)) \models \neg p_{12} \wedge \neg p_{22}, \ldots, \neg p_{(b-2)2}$ i.e. second column of the upper-left corner triangle.
  1. $\mathcal{P}_n^b \wedge sbp_{\sigma(1,2)} \models \neg p_{12}$: Let $\sigma = (p_{12}, p_{22}), (p_{13}, p_{23}) \subset \sigma(1,2)$. $sbp_\sigma = (p_{12} \leqslant p_{22}) \wedge (p_{12} = p_{22}) \rightarrow (p_{13} \leqslant p_{23})$. The CNF formula associated to $sbp(\sigma)$ is $(1)(\neg p_{12} \vee p_{22})$, $(2)(p_{12} \vee p_{22} \vee \neg p_{13} \vee p_{23})$, $(3)(\neg p_{12} \vee \neg p_{22} \vee \neg p_{13} \vee p_{23})$. The clause (3) is subsumed by the binary clause $c = (\neg p_{12} \vee \neg p_{22}) \in \mathcal{P}_n^b$ (see formula (8)). $\eta[p_{22}, (1), c] = \neg p_{12}$. By propagating $\neg p_{12}$, we eliminate p_{12} from the second row of the matrix and we satisfy all the binary clauses of (8) involving $\neg p_{12}$.
  2. Using the same reasoning, we can prove that $\mathcal{P}_n^b \wedge sbp(\mathcal{S}ym(\mathcal{P}_n^b)) \models \neg p_{22} \wedge, \ldots, \neg p_{(b-2)2}$
- Following this resolution process between the clauses of $sbp(\mathcal{S}ym(\mathcal{P}_n^b))$ and $\mathcal{P}_n^b$, we deduce that all the literals belonging to the upper-left corner triangle can be assigned to $false$.

Eliminating The Lower-Right Corner Triangle Let us now describe how the literals involved in the lower-right corner triangle can be eliminated. This second reduction phase is obtained in a similar way, but with different resolution steps. For this second reduction, we start from the formula $\mathcal{P}_n^b$ with the literals of the upper-left corner triangle assigned to $false$. It is achieved by the following resolution process.

- $u\mathcal{P}_n^b \wedge sbp(\mathcal{S}ym(\mathcal{P}_n^b)) \models \neg p_{2n} \wedge \neg p_{3n}, \ldots, \neg p_{bn}$ i.e. we eliminate the variables of the last column of the lower-right corner triangle.

1. $u\mathcal{P}_n^b \wedge sbp(\mathcal{S}ym(\mathcal{P}_n^b)) \models \neg p_{2n}$: By resolution between the positive clause $(p_{1b} \vee, \ldots, \vee p_{1(n-1)} \vee p_{1n})$ and the negative binary clause $(\neg p_{1n} \vee \neg p_{2n})$, we obtain the clause $r_1 = (p_{1b} \vee, \ldots, \vee p_{1(n-1)} \vee \neg p_{2n})$. A second resolution step between r_1 and $(p_{2(b-1)} \vee, \ldots, \vee p_{2(n-1)} \vee p_{2n})$, allows us to deduce $r_2 = (p_{1b} \vee, \ldots, \vee p_{1(n-2)} \vee p_{1(n-1)} \vee p_{2(b-1)} \vee, \ldots, \vee p_{2(n-1)})$. To eliminate the first $n-1$ literals from r_2, we exploit $sbp_{\sigma(1,i)}$ with $2 \leqslant i \leqslant b$. The literal $p_{1(n-1)}$ is eliminated from r_2 by resolution between the clause $s_1 = (p_{2(b-1)} \vee p_{1b} \vee p_{2b} \vee \ldots, \vee p_{1(n-2)} \vee p_{2(n-2)} \vee \neg p_{1(n-1)} \vee p_{2(n-1)}) \in sbp_{\sigma(1,2)}$ and r_2. The clause $s_1 \in (p_{11} = p_{21}) \wedge, \cdots \wedge (p_{1(b-1)} = p_{2(b-1)}) \wedge, \ldots, \wedge (p_{1(n-2)} = p_{2(n-2)}) \rightarrow p_{1(n-1)} \leqslant p_{2(n-1)})$. All the literals $\{p_{11}, p_{21}, \ldots, p_{1(b-1)}\}$ are false (upper-left corner triangle). We have the resolvent $\eta[p_{1(n-1)}, r_2, s_1] = r_3 = (p_{1b} \vee, \ldots, \vee p_{1(n-2)} \vee p_{2(b-1)} \vee, \ldots, \vee p_{2(n-1)})$. Now, we show how $p_{1(n-2)}$ can be eliminated from r_3. Let $s_2 = (p_{2(b-1)} \vee p_{1b} \vee p_{2b} \vee \ldots, \vee p_{1(n-3)} \vee p_{2(n-3)} \vee \neg p_{1(n-2)} \vee p_{2(n-1)}) \in sbp_{\sigma(1,2)}$. We obtain the resolvent $\eta[p_{1(n-2)}, r_3, s_2] = r_4 = (p_{1b} \vee, \ldots, \vee p_{1(n-3)} \vee p_{2(b-1)} \vee, \ldots, \vee p_{2(n-1)})$. The remaining literals $\{p_{1b}, \ldots, p_{1(n-3)}\}$ from r_4 can be eliminated by iterating the same reasoning. Then we deduce the clause $(p_{2(b-1)} \vee, \ldots, \vee p_{2(n-1)})$, the clause in brackets (second row of the matrix). The previous reasoning allows us to eliminate the single occurrence of the positive literal p_{2n}. Then the literal $\neg p_{2n}$ becomes pure. Consequently, the binary clauses from (8) containing $\neg p_{2n}$ can be eliminated.
2. To eliminate the literals $p_{3n}, \ldots,$ and p_{bn}, we use exactly the same reasoning with $sbp_{\sigma(1,3)}, \ldots, sbp_{\sigma(1,b)}$ respectively.

- Following the above process,with $sbp_{\sigma(2,3)}, \ldots,$ and $sbp_{\sigma(2,b)}$, we eliminate the literals of the second column of the lower-right corner triangle $p_{3(n-1)}, \ldots, p_{b(n-1)}$ respectively. We iterate this process until eliminating the last column of the triangle made only of a single literal $p_{b(n-(b-2))}$.

To end the reduction of $\mathcal{P}_n^b$ to $ph\mathcal{P}_n^b$, we only need to show how the formula (11) can be derived and used to substitute the set of binary clauses (8). To achieve this, we use a similar but slightly different process involving tricky and non trivial resolution steps on the formula obtained by the previous reductions. This last reduction step is more complicated than the two previous reductions of upper-left and lower-right corner triangles. It involves both symmetry breaking predicates, redundant clauses expressing that a pigeon can not be put in two different holes, and additional blocked clauses. We omit this last reduction step and we prove in the sequel that our $ph\mathcal{P}_n^b$ encoding of the cardinality constraint is sound and preserves unit propagation.

We have shown how a resolution process involving clauses of the original formulation $\mathcal{P}_n^b$ and clauses from symmetry breaking predicates $sbp(\mathcal{S}ym(\mathcal{P}_n^b))$ can be very useful in reducing the size of the formula. This contrast with the usual way to break symmetries where the SBP's are conjunctively added to the original formula.

It is important to note that except for the elimination of the upper-left triangle, the other reductions involve more complicated reasoning and can not be obtained by a simple reasoning on the CSP formulation using the lex-leader constraints (Puget 2005; Walsh 2007) or row symmetries (Flener et al. 2002; Katsirelos, Narodytska, and Walsh 2010).

Soundness and Unit Propagation

Soundness In the following two propositions, we show that our encoding of constraint cardinality is sound.

Proposition 2. *If ρ is a model of $ph\mathcal{P}_n^b$ then ρ is a model of $\sum_{i=1}^n x_i \geqslant b$.*

Proof. Let us assume that ρ is not a model of $\sum_{i=1}^n x_i \geqslant b$. Then, there exist $n-b+1$ distinct integers $i_1 < \cdots < i_{n-b+1}$ such that $\rho(x_{i_1}) = \cdots = \rho(x_{i_{n-b+1}}) = 0$. Moreover, using unit propagation in (9), we have for all propositional variables of the form $p_{k(i_j)}$ for $j = 1, \ldots, n-b+1$, $\rho(p_{k(i_j)}) = 0$ holds. If $i_{n-b+1} = i_{n-b} + 1 = \cdots = i_1 + n - b$, then one can see that there exists a clause in (10) which is not satisfied by ρ and we get a contradiction. Otherwise, using unit propagation in (11), we deduce that there exist $n-b+1$ distinct integers $i'_1 < \cdots < i'_{n-b+1}$ such that $i'_{n-b+1} = i'_{n-b} + 1 = \cdots = i'_1 + n - b$ and $\rho(p_{(b-i'_1+1)(i'_j)}) = 0$ for $j = 1, \ldots, n-b+1$. Thus, the clause $p_{(b-i'_1+1)(i'_1)} \vee \cdots \vee p_{(b-i'_1+1)(i'_{n-b+1})}$ in (10) is not satisfied by ρ and we get also a contradiction. □

Proposition 3. *If ρ is a model of $\sum_{i=1}^n x_i \geqslant b$, then there exists a model ρ' of $ph\mathcal{P}_n^b$ such that for all $i \in \{1, \ldots, n\}$, $\rho(x_i) = \rho'(x_i)$.*

Proof. Let us consider that $\rho(\sum_{i=1}^n x_i) = l \geqslant b$. We know that there exists a set S of l propositional variables $\{p_{k_1 i_1}, \ldots, p_{k_l i_l}\}$ such that: for all $m, m' \in \{1, \ldots, l\}$, $\rho(x_{i_m}) = 1$, and $i_m \neq i_{m'}$ when $m \neq m'$; and $\{1, \ldots, b\} \subseteq \{k_1, \ldots, k_l\}$. Let us now define ρ' on the propositional variables of $ph\mathcal{P}_n^b$ not in $\{x_1, \ldots, x_n\}$ as follows:

$$\rho'(p) = \begin{cases} 1 & \text{if } p \in S \\ 0 & \text{otherwise} \end{cases}$$

One can see that $\rho'(9) = 1$. Moreover, using the fact that $\{1, \ldots, b\} \subseteq \{k_1, \ldots, k_l\}$, we get $\rho'(10) = 1$. Finally, since for all $m, m' \in \{1, \ldots, l\}$ we have $i_m \neq i_{m'}$ when $m \neq m'$, we deduce that $\rho'(11) = 1$. □

Unit Propagation Let us now prove that $ph\mathcal{P}_n^b$ ensures generalized arc consistency by unit propagation, one of the most important properties for the efficiency of the encoding.

Proposition 4 (Unit propagation). *Let ρ be a model of $ph\mathcal{P}_n^b$ assigning 0 to the elements of a set $S = \{x_{i_1}, \ldots, x_{i_{n-b}}\}$ of $n-b$ propositional variables included in $X = \{x_1, \ldots, x_n\}$. Unit propagation is sufficient to deduce that for all variable $x \in X \backslash S$, $\rho(x) = 1$.*

Proof. We assume without loss of generality that $i_1 < \cdots < i_{n-b}$. Let us note that we have the following clauses in (11):

$$p_{(b-k+1)k} \vee \neg p_{(b-k)(k+1)}, \quad k = 1, \ldots, b-1 \tag{13}$$

If $i_{n-b} = i_{n-b-1} + 1 = \cdots = i_1 + n - b - 1$, then, by using unit propagation in (9), (10) and (13), we get for all $j \in \{1, \ldots, i_1 - 1\}$, $\rho(x_j) = 1$. That is mainly because of the fact that one can obtain $\rho(p_{(i_1+1)(i_1-1)}) = 1$ by using unit propagation in (9) and the clause $p_{(i_1+1)(i_1-1)} \vee \cdots \vee p_{(i_1+1)(i_{n-b})}$. Moreover, by using unit propagation in (9), (10) and (11), we get for all $j \in \{i_{n-b} + 1, \ldots, n\}$, $\rho(x_j) = 1$. Indeed, using unit propagation in (10) and (11) allows us to deduce that for all $k \in \{i_1, \ldots, b\}$, $\rho(p_{(b-k+1)(k+n-b)}) = 1$.
Let us now consider the case where $i_1, \cdots, i_{n-b}$ are not immediate successive integers. By using now unit propagation in (9) and (11), we deduce that for all $i \in I_1 = \{b, \ldots, n\} \backslash \{i_1, \cdots, i_{n-b}\}$ which is different from the greatest element l_1 of I_1, $\rho(p_{1i}) = 0$, and consequently $\rho(p_{1l_1}) = 1$. For instance, let us see that by using unit propagation in (9) and (13), we deduce that $\rho(p_{(b-(i_1+k)+1)(i_i+k)}) = 0$ for $k = 1, \ldots, b - i_1$. By using also unit propagation in (11), we obtain that for all $i \in I_2 = (\{b-1, \ldots, n-1\} - \{l_1\}) \backslash \{i_1, \cdots, i_{n-b}\}$ which is different from the greatest element l_2 of I_2, $\rho(p_{2i}) = 0$, and consequently $\rho(p_{2l_2}) = 1$. We proceed similarly until we obtain $\rho(p_{1l_1}) = \rho(p_{2l_2}) = \cdots = \rho(p_{bl_b}) = 1$. Finally, by using unit propagation in (9), we conclude that for all $i \in \{1, \ldots, n\} \backslash \{i_1, \cdots, i_{n-b}\}$, $\rho(x_i) = 1$ holds.

□

At Most One Constraint: A Particular Case

In this section, we consider the at most one constraint $\sum_{i=1}^n x_i \leqslant 1$, a special case of the cardinality constraint. First, the constraint $\sum_{i=1}^n x_i \leqslant 1$ can be equivalently written as $\sum_{i=1}^n \neg x_i \geq (n-1)$. Using our Pigeon-Hole based encoding $ph\mathcal{P}_n^b$, we obtain the following CNF formula:

$$(\neg p_{(n-k)k} \vee \neg x_k) \wedge (\neg p_{(n-k)(k+1)} \vee \neg x_{k+1}), \quad k = 1, \ldots, (n-1) \tag{14}$$

$$(p_{(n-k)k} \vee p_{(n-k)(k+1)}), \quad k = 1, \ldots, (n-1) \tag{15}$$

$$p_{(n-k)k} \vee \neg p_{(n-(k+1))(k+1)}, \ k = 1, \ldots, (n-2) \tag{16}$$

Note that $p_{(n-k)(k+1)}, k \in \{1, \ldots, n-1\}$ appear negatively only in the clause $(\neg p_{(n-k)(k+1)} \vee \neg x_{k+1})$. Using an additional blocked clause $(p_{(n-k)(k+1)} \vee x_{k+1})$ we express that $p_{(n-k)(k+1)}$ is equivalent to $\neg x_{k+1}$. After substitution of $p_{(n-k)(k+1)}$ by $\neg x_{k+1}$ and simplification, the new formula is written as :

$$(\neg p_{(n-k)k} \vee \neg x_k), \quad k = 1, \ldots, (n-1) \tag{17}$$

$$(p_{(n-k)k} \vee \neg x_{k+1}), \quad k = 1, \ldots, (n-1) \tag{18}$$

$$p_{(n-k)k} \vee \neg p_{(n-(k+1))(k+1)}, \ k = 1, \ldots, (n-2) \tag{19}$$

Note that substitution allows us to reduce the number of variables from $2 \times n$ additional variables to n and from $4 \times n$ clauses to $3 \times n$. As the number of additional variables is equal to n we can simplify our notation. For example $p_{(n-k)k}$ can be written simply as p_k leading to the following simple encoding:

$$(\neg x_1 \vee p_1) \wedge (\neg x_n \vee p_{n-1}) \bigwedge_{1<i<n} ((\neg x_i \vee p_i) \wedge (\neg p_{i-1} \vee p_i) \wedge (\neg x_i \vee \neg p_{i-1})) \tag{20}$$

As we can see, the resulting formula (20) is the same than the one derived using sequential counter (Sinz 2005; Silva and Lynce 2007).

Theoretical Comparison With Other Encodings

In this section, we compare our Pigeon-Hole based encodings with several well-known state-of-the-art encodings. In Table 1, we give a comparison in term of number of variables and clauses. We also mention if the encoding can be decided by unit propagation, or by search. For our comparison, we consider the cardinality constraint $\sum_{i=1}^n x_i \leqslant b$ as it is the most used in the literature. This cardinality constraint is equivalent to $\sum_{i=1}^n \neg x_i \leqslant n - b$. The complexity of our encoding is the same for both kinds of cardinality constraints.

The first naive approach (without auxiliary variables) for encoding the cardinality constraint is exponential in the worst case. Most of the proposed encodings aims to both reduce the size of the encoding and to improve its propagation capabilities. Some of the most known encodings are summarized in Table 1. As we can see, the best current available encoding is clearly the cardinality network encoding proposed recently in (Asín et al. 2011). Comparatively, our $ph\mathcal{P}_n^b$ is clearly competitive both in size and efficiency. As we mentioned previously, symmetries among the variables x_i are not considered in our encoding, we believe that further improvements might be obtained.

Encoding	# Clauses	# Variables	Decided
Sequential unary counter ($LT_{SEQ}^{n,b}$)(Sinz 2005)	$\mathcal{O}(b \times n)$	$\mathcal{O}(b \times n)$	UP
Parallel binary counter ($LT_{SEQ}^{n,b}$)(Sinz 2005)	$7n - 3log(n) - 6$	$2n - 2$	Search
Totalizer (Bailleux and Boufkhad 2003)	$\mathcal{O}(b \times n)$	$\mathcal{O}(n \times log_2(n))$	UP
Buttner & Rintanen (Büttner and Rintanen 2005)	$\mathcal{O}(b^2 \times n)$	$\mathcal{O}(n \times log_2(n))$	UP
Sorting Network (Eén and Sörensson 2006)	$\mathcal{O}(n \times log_2^2(n))$	$\mathcal{O}(n \times log_2^2(n))$	UP
Cardinality Network (Asín et al. 2011)	$\mathcal{O}(n \times log_2^2(b))$	$\mathcal{O}(n \times log_2^2(b))$	UP
Warners (Warners 1996)	$8n$	$2n$	Search
$\mathcal{P}_n^b$	$\mathcal{O}(b \times n)$	$\mathcal{O}(b^2 \times n)$	Search
$ph\mathcal{P}_n^b$	$\mathcal{O}(b \times (n-b))$	$\mathcal{O}(b \times (n-b))$	UP

Table 1: Comparison of CNF Encodings of $\sum_{i=1}^n x_i \leqslant b$

Conclusion and Future Works

In this paper, we proposed a new and efficient Pigeon-Hole based encoding of cardinality constraints to CNF. Our encoding is competitive as it derives a CNF formula with $O(b(n-b))$ variables and clauses. As the obtained CNF formula is reverse Horn, unit propagation is sufficient for deciding its satisfiability. The originality of our proposed approach rises in the use for the first time of the Pigeon-Hole principle to naturally model counting constraints. We have also shown that when resolution is applied between clauses of the original Pigeon-Hole encoding and symmetry breaking predicates, one can achieve interesting reduction and improvements. This opens a promising perspective on how to extend the reasoning applied in this paper to other kinds of constraints (e.g. global constraints). The methodology presented in this paper can also be applied to reduce the size of arbitrary CNF formulae using symmetry breaking predicates not as additional constraints but only in the resolution-based reduction process. The generalization of our reasoning to encode general pseudo Boolean constraint to CNF is also a short term perspective. Finally, we plan to conduct an experimental evaluation of our Pigeon-Hole based encoding w.r.t. the well-known encodings.

Acknowledgments

We are grateful to the French ANR Agency for supporting this work under the TUPLES project "Tractability for Understanding and Pushing forward the Limits of Efficient Solvers". We thank the anonymous reviewers for their comments that helped us improve this paper.

References

Aloul, F. A.; Ramani, A.; Markov, I. L.; and Sakallah, K. A. 2003. Solving difficult instances of boolean satisfiability in the presence of symmetry. *IEEE Trans. on CAD of Integrated Circuits and Systems* 22(9):1117–1137.

Aloul, F. A.; Sakallah, K. A.; and Markov, I. L. 2006. Efficient symmetry breaking for boolean satisfiability. *IEEE Trans. Computers* 55(5):549–558.

Asín, R.; Nieuwenhuis, R.; Oliveras, A.; and Rodríguez-Carbonell, E. 2009. Cardinality networks and their applications. In *12th International Conference on Theory and Applications of Satisfiability Testing (SAT 2009)*, 167–180.

Asín, R.; Nieuwenhuis, R.; Oliveras, A.; and Rodríguez-Carbonell, E. 2011. Cardinality networks: a theoretical and empirical study. *Constraints* 16(2):195–221.

Bailleux, O., and Boufkhad, Y. 2003. Efficient cnf encoding of boolean cardinality constraints. In *9th International Conference on Principles and Practice of Constraint Programming (CP 2003)*, 108–122.

Bailleux, O.; Boufkhad, Y.; and Roussel, O. 2009. New encodings of pseudo-boolean constraints into cnf. In *SAT'2009*, 181–194.

Benhamou, B., and Sais, L. 1992. Theoretical study of symmetries in propositional calculus and applications. In *11th International Conference on Automated Deduction (CADE'1992)*, volume 607 of *Lecture Notes in Computer Science*, 281–294. Springer.

Benhamou, B., and Sais, L. 1994. Tractability through symmetries in propositional calculus. *Journal of Automated Reasoning* 12(1):89–102.

Büttner, M., and Rintanen, J. 2005. Satisfiability planning with constraints on the number of actions. In Biundo, S.; Myers, K. L.; and Rajan, K., eds., *ICAPS*, 292–299. AAAI.

Cook, S. A. 1976. A short proof of the pigeon hole principle using extended resolution. *SIGACT News* 8(4):28–32.

Crawford, J. M.; Ginsberg, M. L.; Luks, E. M.; and Roy, A. 1996. Symmetry-breaking predicates for search problems. In *KR*, 148–159.

Crawford, J. 1992. A theorical analysis of reasoning by symmetry in first order logic. In *Proceedings of Workshop on Tractable Reasonning, AAAI*, 17–22.

Eén, N., and Sörensson, N. 2006. Translating pseudo-boolean constraints into sat. *JSAT* 2(1-4):1–26.

Flener, P.; Frisch, A. M.; Hnich, B.; Kiziltan, Z.; Miguel, I.; Pearson, J.; and Walsh, T. 2002. Breaking row and column symmetries in matrix models. In *8th International Conference on Principles and Practice of Constraint Programming (CP'2002)*, 462–476. London, UK, UK: Springer-Verlag.

Gent, I. P.; Kelsey, T.; Linton, S.; McDonald, I.; Miguel, I.; and Smith, B. M. 2005. Conditional symmetry breaking. In *11th International Conference on Principles and Practice of Constraint Programming (CP 2005)*, 256–270.

Gent, I. P.; Petrie, K. E.; and Puget, J.-F. 2006. Chapter 10 symmetry in constraint programming. In F. Rossi, P. v. B., and Walsh, T., eds., *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*. Elsevier. 329 – 376.

Katsirelos, G.; Narodytska, N.; and Walsh, T. 2010. On the complexity and completeness of static constraints for breaking row and column symmetry. In *16th International Conference on Principles and Practice of Constraint Programming (CP'2010)*, 305–320.

Krishnamurthy, B. 1985. Shorts proofs for tricky formulas. *Acta Informatica* 22:253–275.

Kullmann, O. 1997. On a generalization of extended resolution. *Discrete Applied Mathematics* 34:73–95.

Puget, J. 1993. On the satisfiability of symmetrical constraint satisfaction problems. In *proceedings of ISMIS*, 350–361.

Puget, J.-F. 2005. Breaking symmetries in all different problems. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI'2005)*, 272–277.

Silva, J. P. M., and Lynce, I. 2007. Towards robust cnf encodings of cardinality constraints. In *13th International Conference on Principles and Practice of Constraint Programming (CP 2007)*, 483–497.

Sinz, C. 2005. Towards an optimal cnf encoding of boolean cardinality constraints. In *11th International Conference*

on Principles and Practice of Constraint Programming (CP 2005), 827–831.

Tseitin, G. 1968. On the complexity of derivations in the propositional calculus. In Slesenko, H., ed., *Structures in Constructives Mathematics and Mathematical Logic, Part II*, 115–125.

Walsh, T. 2006. General symmetry breaking constraints. In *12th International Conference on Principles and Practice of Constraint Programming (CP 2006)*, 650–664.

Walsh, T. 2007. Breaking value symmetry. In *13th International Conference on Principles and Practice of Constraint Programming (CP'2007)*, 880–887.

Warners, J. P. 1996. A linear-time transformation of linear inequalities into conjunctive normal form. *Information Processing Letters*.