A Lower Bound on DNNF Encodings of Pseudo-Boolean Constraints

Alexis de Colnet[(✉)]

CRIL, CNRS & Univ Artois, Lens, France
decolnet@cril.fr

Abstract. Two major considerations when encoding pseudo-Boolean (PB) constraints into SAT are the size of the encoding and its propagation strength, that is, the guarantee that it has a good behaviour under unit propagation. Several encodings with propagation strength guarantees rely upon prior compilation of the constraints into DNNF (decomposable negation normal form), BDD (binary decision diagram), or some other sub-variants. However it has been shown that there exist PB-constraints whose ordered BDD (OBDD) representations, and thus the inferred CNF encodings, all have exponential size. Since DNNFs are more succinct than OBDDs, preferring encodings via DNNF to avoid size explosion seems a legitimate choice. Yet in this paper, we prove the existence of PB-constraints whose DNNFs all require exponential size.

Keywords: PB constraints · Knowledge compilation · DNNF

1 Introduction

Pseudo-Boolean (PB) constraints are Boolean functions over 0/1 Boolean variables $x_1, \ldots, x_n$ of the form $\sum_{i=1}^{n} w_i x_i$ 'op' θ where the w_i are integer weights, θ is an integer threshold and 'op' is a comparison operator $<, \leq, >$ or $\geq$. PB-constraints have been studied extensively under different names (e.g. threshold functions [14], Knapsack constraints [13]) due to their omnipresence in many domains of AI and their wide range of practical applications [3,7,9,15,21].

One way to handle PB-constraints in a constraint satisfaction problem is to translate them into a CNF formula and feed it to a SAT solver. The general idea is to generate a CNF, possibly introducing auxiliary Boolean variables, whose restriction to variables of the constraint is equivalent to the constraint. Two major considerations here are the size of the CNF encoding and its propagation strength. One wants, on the one hand, to avoid the size of the encoding to explode, and on the other hand, to guarantee a good behaviour of the SAT instance under unit propagation – a technique at the very core of SAT solving. Desired propagation strength properties are, for instance, generalized arc consistency (GAC) [4] or propagation completeness (PC) [6]. Several encodings to CNF follow the same two-steps method: first, each constraint is represented in a compact form such as BDD (Binary Decision Diagram) or DNNF (Decomposable Negation Normal Form).

Second, the compact forms are turned into CNFs using Tseitin or other transformations. The SAT instance is the conjunction of all obtained CNFs. It is worth mentioning that there are GAC encodings of PB-constraints into polynomial size CNFs that do not follow this two-steps method [5]. However no similar result is known for PC encodings. PC encodings are more restrictive that GAC encodings and may be obtained via techniques requiring compilation to DNNF [17]. Thus the first step is a knowledge compilation task.

Knowledge compilation studies different representations for knowledge [10, 19] under the general idea that some representations are more suitable than others when solving specific reasoning problems. One observation that has been made is that the more reasoning tasks can be solved efficiently with particular representations, the larger these representations get in size. In the context of constraint encodings to SAT, the conversion of compiled forms to CNFs does not reduce the size of the SAT instance, therefore it is essential to control the size of the representations obtained by knowledge compilation.

Several representations have been studied with respect to different encoding techniques with the purpose of determining which properties of representations are sufficient to ensure propagation strength [1, 2, 11, 12, 16, 17]. Popular representations in this context are DNNF and BDD and their many variants: deterministic DNNF, smooth DNNF, OBDD... As mentioned above, a problem occurring when compiling a constraint into such representations is that exponential space may be required. Most notably, it has been shown in [2, 14] that some PB-constraints can only be represented by OBDDs whose size is exponential in $\sqrt{n}$, where n is the number of variables. Our contribution is the proof of the following theorem where we lift the statement from OBDD to DNNF.

Theorem 1. *There is a class of PB-constraints $\mathcal{F}$ such that for any constraint $f \in \mathcal{F}$ on n^2 variables, any DNNF representation of f has size $2^{\Omega(n)}$.*

Since DNNFs are exponentially more succinct than OBDDs [10], our result is a generalisation of the result in [2, 14]. The class $\mathcal{F}$ is similar to that used in [2, 14], actually the only difference is the choice of the threshold for the PB-constraints. Yet, adapting proofs given in [2, 14] for OBDD to DNNF is not straightforward, thus our proof of Theorem 1 bears very little resemblance.

It has been shown in [18] that there exist sets of PB-constraints such that the whole *set* (so a conjunction of PB-constraints) requires exponential size DNNF to represent. Our result is a generalisation to *single* PB-constraints.

2   Preliminaries

Conventions of Notation. Boolean variables are seen as variables over $\{0, 1\}$, where 0 and 1 represent *false* and *true* respectively. Via this $0/1$ representation, Boolean variables can be used in arithmetic expressions over $\mathbb{Z}$. For notational convenience, we keep the usual operators $\neg$, $\vee$ and $\wedge$ to denote, respectively, the negation, disjunction and conjunction of Boolean variables or functions. Given X a set of n Boolean variables, assignments to X are seen as vectors in $\{0, 1\}^n$.

Single Boolean variables are written in plain text (x) while assignments to several variables are written in bold ($\mathbf{x}$). We write $\mathbf{x} \leq \mathbf{y}$ when the vector $\mathbf{y}$ dominates $\mathbf{x}$ element-wise. We write $\mathbf{x} < \mathbf{y}$ when $\mathbf{x} \leq \mathbf{y}$ and $\mathbf{x} \neq \mathbf{y}$. In this framework, a Boolean function f over X is a mapping from $\{0,1\}^n$ to $\{0,1\}$. f is said to *accept* an assignment $\mathbf{x}$ when $f(\mathbf{x}) = 1$, then $\mathbf{x}$ is called a *model* of f. The function is *monotone* if for any model $\mathbf{x}$ of f, all $\mathbf{y} \geq \mathbf{x}$ are models of f as well. The set of models of f is denoted $f^{-1}(1)$. Given f and g two Boolean functions over X, we write $f \leq g$ when $f^{-1}(1) \subseteq g^{-1}(1)$. We write $f < g$ when the inclusion is strict.

Pseudo-Boolean Constraints. *Pseudo-Boolean (PB) constraints* are inequalities the form $\sum_{i=1}^n w_i x_i$ 'op' θ where the x_i are 0/1 Boolean variables, the w_i and θ are integers, and 'op' is one of the comparison operator $<, \leq, >$ or $\geq$. A PB-constraint is associated with a Boolean function whose models are exactly the assignments to $\{x_1, \ldots, x_n\}$ that satisfy the inequality. For simplicity we directly consider PB-constraints as Boolean functions – although the same function may represent different constraints – while keeping the term "constraints" when referring to them. In this paper, we restrict our attention to PB-constraints where 'op' is $\geq$ and all weights are positive integers. Note that such PB-constraints are monotone Boolean functions. Given a sequence of positive integer weights $W = (w_1, \ldots, w_n)$ and an integer threshold θ, we define the function $w : \{0,1\}^n \to \mathbb{N}$ that maps any assignment to its weight by $w(\mathbf{x}) = \sum_{i=1}^n w_i x_i$. With these notations, a PB-constraint over X for a given pair (W, θ) is a Boolean function whose models are exactly the $\mathbf{x}$ such that $w(\mathbf{x}) \geq \theta$.

Example 1. Let $n = 5$, $W = (1, 2, 3, 4, 5)$ and $\theta = 9$. The PB-constraint for (W, θ) is the Boolean function whose models are the assignments such that $\sum_{i=1}^5 i x_i \geq 9$. E.g. $\mathbf{x} = (0, 1, 1, 0, 1)$ is a model of weight $w(\mathbf{x}) = 10$.

For notational clarity, given any subset $Y \subseteq X$ and denoting $\mathbf{x}|_Y$ the restriction of $\mathbf{x}$ to variables of Y, we overload w so that $w(\mathbf{x}|_Y)$ is the sum of weights activated by variables of Y set to 1 in $\mathbf{x}$.

Decomposable NNF. A circuit in *negation normal form* (NNF) is a single output Boolean circuit whose inputs are Boolean variables and their complements, and whose gates are fanin-2 AND and OR gates. The *size* of the circuit is the number of its gates. We say that an NNF is *decomposable* (DNNF) if for any AND gate, the two sub-circuits rooted at that gate share no input variable, i.e., if x or $\neg x$ is an input of the circuit rooted at the left input of the AND gate, then neither x nor $\neg x$ is an input of the circuit rooted at the right input, and vice versa. A Boolean function f is *encoded* by a DNNF D if the assignments of variables for which the output of D is 1 (*true*) are exactly the models of f.

Rectangle Covers. Let X be a finite set of Boolean variables and let $\Pi = (X_1, X_2)$ be a partition of X (i.e., $X_1 \cup X_2 = X$ and $X_1 \cap X_2 = \emptyset$). A *rectangle r* with respect to Π is a Boolean function over X defined as the conjunction of two functions ρ_1 and ρ_2 over X_1 and X_2 respectively. Π is called the *partition* of r. We say that the partition and the rectangle are *balanced* when $\frac{|X|}{3} \leq |X_1| \leq \frac{2|X|}{3}$ (thus the same holds for X_2). Whenever considering a partition (X_1, X_2), we use

for any assignment $\mathbf{x}$ to X the notations $\mathbf{x}_1 \coloneqq \mathbf{x}|_{X_1}$ and $\mathbf{x}_2 \coloneqq \mathbf{x}|_{X_2}$. And for any two assignments $\mathbf{x}_1$ and $\mathbf{x}_2$ to X_1 and X_2, we note $(\mathbf{x}_1, \mathbf{x}_2)$ the assignment to X whose restrictions to X_1 and X_2 are $\mathbf{x}_1$ and $\mathbf{x}_2$. Given f a Boolean function over X, a *rectangle cover* of f is a disjunction of rectangles over X, possibly with different partitions, equivalent to f. The *size* of a rectangle cover is the number of its rectangles. A cover is called *balanced* if all its rectangles are balanced.

Example 2. Going back to Example 1, consider the partition $X_1 \coloneqq \{x_1, x_3, x_4\}$, $X_2 \coloneqq \{x_2, x_5\}$ and define $\rho_1 \coloneqq x_3 \wedge x_4$ and $\rho_2 \coloneqq x_2 \vee x_5$. Then $r \coloneqq \rho_1 \wedge \rho_2$ is a rectangle w.r.t. this partition that accepts only models of the PB-constraint from Example 1. Thus it can be part of a rectangle cover for this constraint.

Any function f has at least one balanced rectangle cover as one can create a balanced rectangle accepting exactly one chosen model of f. We denote by $C(f)$ the size of the smallest balanced rectangle cover of f. The following result from [8] links $C(f)$ to the size of any DNNF encoding f.

Theorem 2. *Let D be a DNNF encoding a Boolean function f. Then f has a balanced rectangle cover of size at most the size of D.*

Theorem 2 reduces the problem of finding lower bounds on the size of DNNFs encoding f to that of finding lower bounds on $C(f)$.

3 Restriction to Threshold Models of PB-Constraints

The strategy to prove Theorem 1 is to find a PB-constraint f over n variables such that $C(f)$ is exponential in $\sqrt{n}$ and then use Theorem 2. We first show that we can restrict our attention to covering particular models of f with rectangles rather than the whole function. In this section X is a set of n Boolean variables and f is a PB-constraint over X. Recall that we only consider constraints of the form $\sum_{i=1}^{n} w_i x_i \geq \theta$ where the w_i and θ are positive integers.

Definition 1. *The* threshold models *of f are the models $\mathbf{x}$ such that $w(\mathbf{x}) = \theta$.*

Threshold models should not be confused with minimal models (or minimals).

Definition 2. *A* minimal *of f is a model $\mathbf{x}$ such that no $\mathbf{y} < \mathbf{x}$ is a model of f.*

For a monotone PB-constraint, a minimal model is such that its sum of weights drops below the threshold if we remove any element from it. Any threshold model is minimal, but not all minimals are threshold models. There even exist constraints with no threshold models (e.g. take even weights and an odd threshold) while there always are minimals for satisfiable constraints.

Example 3. The minimals of the PB-constraint from Example 1 are $(0, 0, 0, 1, 1)$, $(0, 1, 1, 1, 0)$, $(1, 0, 1, 0, 1)$ and $(0, 1, 1, 0, 1)$. The first three are threshold models.

Let f^* be the Boolean function whose models are exactly the threshold models of f. In the next lemma, we prove that the smallest rectangle cover of f^* has size at most $C(f)$. Thus, lower bounds on $C(f^*)$ are also lower bounds on $C(f)$.

Lemma 1. *Let f^* be the Boolean function whose models are exactly the threshold models of f. Then $C(f) \geq C(f^*)$.*

Proof. Let $r := \rho_1 \wedge \rho_2$ be a balanced rectangle with $r \leq f$ and assume r accepts some threshold models. Let $\Pi := (X_1, X_2)$ be the partition of r. We claim that there exist two integers θ_1 and θ_2 such that $\theta_1 + \theta_2 = \theta$ and, for any threshold model $\mathbf{x}$ accepted by r, there is $w(\mathbf{x_1}) = \theta_1$ and $w(\mathbf{x_2}) = \theta_2$. To see this, assume by contradiction that there exists another partition $\theta = \theta_1' + \theta_2'$ of θ such that some other threshold model $\mathbf{y}$ with $w(\mathbf{y_1}) = \theta_1'$ and $w(\mathbf{y_2}) = \theta_2'$ is accepted by r. Then either $w(\mathbf{x_1}) + w(\mathbf{y_2}) < \theta$ or $w(\mathbf{y_1}) + w(\mathbf{x_2}) < \theta$, but since $(\mathbf{x_1}, \mathbf{y_2})$ and $(\mathbf{y_1}, \mathbf{x_2})$ are also models of r, r would accept a non-model of f, which is forbidden. Now let ρ_1^* (resp. ρ_2^*) be the function whose models are exactly the models of ρ_1 (resp. ρ_2) of weight θ_1 (resp. θ_2). Then $r^* := \rho_1^* \wedge \rho_2^*$ is a balanced rectangle whose models are exactly the threshold models accepted by r.

Now consider a balanced rectangle cover of f of size $C(f)$. For each rectangle r of the cover, if r accepts no threshold model then discard it, otherwise construct r^*. The disjunction of these new rectangles is a balanced rectangle cover of f^* of size at most $C(f)$. Therefore $C(f) \geq C(f^*)$. □
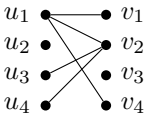
4   Reduction to Covering Maximal Matchings of $K_{n,n}$

We define the class of hard PB-constraints for Theorem 1 in this section. Recall that for a hard constraint f, our aim is to find an exponential lower bound on $C(f)$. We will show, using Lemma 1, that the problem can be reduced to that of covering all maximal matchings of the complete $n \times n$ bipartite graph $K_{n,n}$ with rectangles. In this section, X is a set of n^2 Boolean variables. For presentability reasons, assignments to X are written as $n \times n$ matrices. Each variable $x_{i,j}$ has the weight $w_{i,j} := (2^i + 2^{j+n})/2$. Define the matrix of weights $W := (w_{i,j} : 1 \leq i, j \leq n)$ and the threshold $\theta := 2^{2n} - 1$. The PB-constraint f for the pair (W, θ) is such that $f(\mathbf{x}) = 1$ if and only if $\mathbf{x}$ satisfies

$$\sum_{1 \leq i,j \leq n} \left(\frac{2^i + 2^{j+n}}{2} \right) x_{i,j} \geq 2^{2n} - 1 \ . \tag{1}$$

Constraints of this form constitute the class of hard constraints of Theorem 1. One may find it easier to picture f writing the weights and threshold as binary numbers of $2n$ bits. Bits of indices 1 to n form the *lower part* of the number and those of indices $n + 1$ to $2n$ form the *upper part*. The weight $w_{i,j}$ is the binary number where the only bits set to 1 are the ith bit of the lower part and the jth bit of the upper part. Thus when a variable $x_{i,j}$ is set to 1, exactly one bit of value 1 is added to each part of the binary number of the sum.
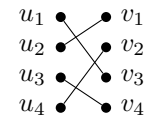
Assignments to X uniquely encode subgraphs of $K_{n,n}$. We denote $U = \{u_1, \ldots, u_n\}$ the nodes of the left side and $V = \{v_1, \ldots, v_n\}$ those of the right side of $K_{n,n}$. The bipartite graph encoded by $\mathbf{x}$ is such that there is an edge between the u_i and v_j if and only if $x_{i,j}$ is set to 1 in $\mathbf{x}$.

Example 4. Take $n = 4$. The assignment $\mathbf{x} = \begin{pmatrix} 1\ 1\ 0\ 1 \\ 0\ 0\ 0\ 0 \\ 0\ 1\ 0\ 0 \\ 0\ 1\ 0\ 0 \end{pmatrix}$ encodes

$$
\begin{array}{ll}
u_1 \bullet & \bullet v_1 \\
u_2 \bullet & \bullet v_2 \\
u_3 \bullet & \bullet v_3 \\
u_4 \bullet & \bullet v_4
\end{array}
$$

Definition 3. *A* maximal matching assignment *(or maximal matching model) is an assignment* $\mathbf{x}$ *to* X *such that*

- *for any* $i \in [n]$, *there is exactly one* k *such that* $x_{i,k}$ *is set to 1 in* $\mathbf{x}$,
- *for any* $j \in [n]$, *there is exactly one* k *such that* $x_{k,j}$ *is set to 1 in* $\mathbf{x}$.

As the name suggests, the maximal matching assignments are those encoding graphs whose edges form a maximal matching of $K_{n,n}$ (i.e., a maximum cardinality matching). One can also see them as encodings for permutations of $[n]$.

Example 5. The maximal matching model $\mathbf{x} = \begin{pmatrix} 0\ 0\ 1\ 0 \\ 1\ 0\ 0\ 0 \\ 0\ 0\ 0\ 1 \\ 0\ 1\ 0\ 0 \end{pmatrix}$ encodes

$$
\begin{array}{ll}
u_1 \bullet & \bullet v_1 \\
u_2 \bullet & \bullet v_2 \\
u_3 \bullet & \bullet v_3 \\
u_4 \bullet & \bullet v_4
\end{array}
$$

For a given $\mathbf{x}$, define $\mathrm{var}_k (\mathbf{x})$ by $\mathrm{var}_k (\mathbf{x}) := \{j \mid x_{k,j} \text{ is set to 1 in } \mathbf{x}\}$ when $1 \le k \le n$ and by $\mathrm{var}_k (\mathbf{x}) := \{i \mid x_{i,k-n} \text{ is set to 1 in } \mathbf{x}\}$ when $n + 1 \le k \le 2n$. $\mathrm{var}_k (\mathbf{x})$ stores the index of variables in $\mathbf{x}$ that directly add 1 to the kth bit of $w(\mathbf{x})$. Note that a maximal matching model is an assignment $\mathbf{x}$ such that $|\mathrm{var}_k (\mathbf{x})| = 1$ for all k. It is easy to see that maximal matching models are threshold models of f: seeing weights as binary numbers of $2n$ bits, for every bit of the sum the value 1 is added exactly once, so exactly the first $2n$ bits of the sum are set to 1, which gives us θ. Note that not all threshold models of f are maximal matching models, for instance the assignment from Example 4 does not encode a maximal matching but one can verify that it is a threshold model. Recall that f^* is the function whose models are the threshold models of f. In the next lemmas, we prove that lower bounds on the size of rectangle covers of the maximal matching models are lower bounds on $C(f^*)$, and a fortiori on $C(f)$.

Lemma 2. *Let* $\Pi := (X_1, X_2)$ *be a partition of* X. *Let* $\mathbf{x} := (\mathbf{x}_1, \mathbf{x}_2)$ *and* $\mathbf{y} := (\mathbf{y}_1, \mathbf{y}_2)$ *be maximal matching assignments. If* $(\mathbf{x}_1, \mathbf{y}_2)$ *and* $(\mathbf{y}_1, \mathbf{x}_2)$ *both have weight* $\theta := 2^{2n} - 1$ *then both are maximal matching assignments.*

Proof. It is sufficient to show that $|\mathrm{var}_k (\mathbf{x}_1, \mathbf{y}_2)| = 1$ and $|\mathrm{var}_k (\mathbf{y}_1, \mathbf{x}_2)| = 1$ for all $1 \le k \le 2n$. We prove it for $(\mathbf{x}_1, \mathbf{y}_2)$ by induction on k. First observe that since $|\mathrm{var}_k (\mathbf{x})| = 1$ and $|\mathrm{var}_k (\mathbf{y})| = 1$ for all $1 \le k \le 2n$, the only possibilities for $|\mathrm{var}_k (\mathbf{x}_1, \mathbf{y}_2)|$ are 0, 1 or 2.

- For the base case $k = 1$, if $|\mathrm{var}_1 (\mathbf{x}_1, \mathbf{y}_2)|$ is even then the first bit of $w(\mathbf{x}_1) + w(\mathbf{y}_2)$ is 0 and the weight of $(\mathbf{x}_1, \mathbf{y}_2)$ is not θ. So $|\mathrm{var}_1 (\mathbf{x}_1, \mathbf{y}_2)| = 1$.
- For the general case $1 < k \le 2n$, assume that $|\mathrm{var}_1 (\mathbf{x}_1, \mathbf{y}_2)| = \cdots = |\mathrm{var}_{k-1} (\mathbf{x}_1, \mathbf{y}_2)| = 1$. So the kth bit of $w(\mathbf{x}_1) + w(\mathbf{y}_2)$ depends only on the parity of $|\mathrm{var}_k (\mathbf{x}_1, \mathbf{y}_2)|$: the kth bit is 0 if $|\mathrm{var}_k (\mathbf{x}_1, \mathbf{y}_2)|$ is even and 1 otherwise. $(\mathbf{x}_1, \mathbf{y}_2)$ has weight θ so $|\mathrm{var}_k (\mathbf{x}_1, \mathbf{y}_2)| = 1$.

The argument applies to $(\mathbf{y}_1, \mathbf{x}_2)$ analogously. $\square$

Lemma 3. *Let f be the PB-constraint (1) and let $\hat{f}$ be the function whose models are exactly the maximal matching assignments. Then $C(f) \geq C(\hat{f})$.*

Proof. By Lemma 1, it is sufficient to prove that $C(f^*) \geq C(\hat{f})$. We already know that $\hat{f} \leq f^*$. Let $r := \rho_1 \wedge \rho_2$ be a balanced rectangle of partition $\Pi := (X_1, X_2)$ with $r \leq f^*$, and assume r accepts some maximal matching assignment. Let $\hat{\rho}_1$ (resp. $\hat{\rho}_2$) be the Boolean function over X_1 (resp. X_2) whose models are the $\mathbf{x}_1$ (resp. $\mathbf{x}_2$) such that there is a maximal matching assignment $(\mathbf{x}_1, \mathbf{x}_2)$ accepted by r. We claim that the balanced rectangle $\hat{r} := \hat{\rho}_1 \wedge \hat{\rho}_2$ accepts exactly the maximal matching models of r. On the one hand, it is clear that all maximal matching models of r are models of $\hat{r}$. On the other hand, all models of $\hat{r}$ are threshold models of the form $(\mathbf{x}_1, \mathbf{y}_2)$, where $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{y}_1, \mathbf{y}_2)$ encode maximal matchings, so by Lemma 2, $\hat{r}$ accepts only maximal matching models of r.

Now consider a balanced rectangle cover of f^* of size $C(f^*)$. For each rectangle r of the cover, if r accepts no maximal matching assignment then discard it, otherwise construct $\hat{r}$. The disjunction of these new rectangles is a balanced rectangle cover of $\hat{f}$ of size at most $C(f^*)$. Therefore $C(f^*) \geq C(\hat{f})$. □

5   Proof of Theorem 1

Theorem 1. *There is a class of PB-constraints $\mathcal{F}$ such that for any constraint $f \in \mathcal{F}$ on n^2 variables, any DNNF encoding f has size $2^{\Omega(n)}$.*

$\mathcal{F}$ is the class of constraints defined in (1). Thanks to Theorem 2 and Lemma 3, the proof boils down to finding exponential lower bounds on $C(\hat{f})$, where $\hat{f}$ is the Boolean function on n^2 variables whose models encode exactly the maximal matchings of $K_{n,n}$ (or equivalently, the permutations of $[n]$). $\hat{f}$ has $n!$ models. The idea is now to prove that rectangles covering $\hat{f}$ must be relatively small, so that covering the whole function requires many of them.

Lemma 4. *Let $\Pi = (X_1, X_2)$ be a balanced partition of X. Let r be a rectangle with respect to Π with $r \leq \hat{f}$. Then $|r^{-1}(1)| \leq n!/\binom{n}{n\sqrt{2/3}}$.*

The function $\hat{f}$ has already been studied extensively in the literature, often under the name PERM_n (for *permutations on $[n]$*), see for instance Chap. 4 of [22] or Sect. 6.2 of [20] where a statement similar to Lemma 4 is established. With Lemma 4 we can give the proof of Theorem 1.

Proof (Theorem 1). Let $\bigvee_{k=1}^{C(\hat{f})} r_k$ be a balanced rectangle cover of $\hat{f}$. We have $\sum_{k=1}^{C(\hat{f})} |r_k^{-1}(1)| \geq |\hat{f}^{-1}(1)| = n!$. Lemma 4 gives us $(C(\hat{f})n!)/\binom{n}{n\sqrt{2/3}} \geq n!$, thus

$$C(\hat{f}) \geq \binom{n}{n\sqrt{2/3}} \geq \left(\frac{n}{n\sqrt{2/3}}\right)^{n\sqrt{2/3}} = \left(\frac{3}{2}\right)^{n\frac{\sqrt{2/3}}{2}} \geq 2^{n\frac{\sqrt{2/3}}{4}} = 2^{\Omega(n)}$$

where we have used $\binom{a}{b} \geq (a/b)^b$ and $3/2 \geq \sqrt{2}$. Using Lemma 3 we get that $C(f) \geq C(\hat{f}) \geq 2^{\Omega(n)}$. Theorem 2 allows us to conclude. □

(a) Balanced partition Π of $K_{4,4}$ (b) Partition of a maximal matching w.r.t. Π
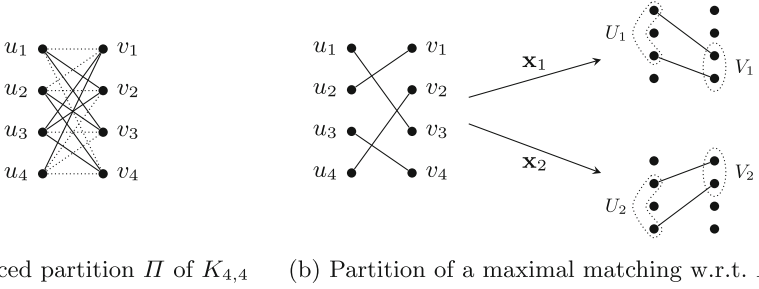
Fig. 1. Partition of maximal matching

All that is left is to prove Lemma 4.

Proof (Lemma 4).

Let $r := \rho_1 \wedge \rho_2$ and $\Pi := (X_1, X_2)$. Recall that $U := \{u_1, \ldots, u_n\}$ and $V := \{v_1, \ldots, v_n\}$ are the nodes from the left and right part of $K_{n,n}$ respectively. Define $U_1 := \{u_i \mid$ there exists $x_{i,l} \in X_1$ such that a model of ρ_1 has $x_{i,l}$ set to 1$\}$ and $V_1 := \{v_j \mid$ there exists $x_{l,j} \in X_1$ such that a model of ρ_1 has $x_{l,j}$ set to 1$\}$. Define U_2 and V_2 analogously (this time using X_2 and ρ_2). Figure 1 illustrates the construction of these sets: Fig. 1a shows a partition Π of the edges of $K_{4,4}$ (full edges in X_1, dotted edges in X_2) and Fig. 1b shows the contribution of a model of r to U_1, V_1, U_2, and V_2 after partition according to Π.

Models of ρ_1 are clearly matchings of $K_{n,n}$. Actually they are matchings between U_1 and V_1 by construction of these sets. We claim that they are maximal. To verify this, observe that $U_1 \cap U_2 = \emptyset$ and $V_1 \cap V_2 = \emptyset$ since otherwise r has a model that is not a matching. Thus if ρ_1 were to accept a non-maximal matching between U_1 and V_1 then r would accept a non-maximal matching between U and V. So ρ_1 accepts only maximal matchings between U_1 and V_1, consequently $|U_1| = |V_1|$. The argument applies symmetrically for V_2 and U_2. We note $k := |U_1|$. It stands that $U_1 \cup U_2 = U$ and $V_1 \cup V_2 = V$ as otherwise r accepts matchings that are not maximal. So $|U_2| = |V_2| = n - k$. We now have $|\rho_1^{-1}(1)| \leq k!$ and $|\rho_2^{-1}(1)| \leq (n-k)!$, leading to $|r^{-1}(1)| \leq k!(n-k)! = n!/\binom{n}{k}$.

Up to k^2 edges may be used to build matchings between U_1 and V_1. Since r is balanced we obtain $k^2 \leq 2n^2/3$. Applying the same argument to U_2 and V_2 gives us $(n-k)^2 \leq 2n^2/3$, so $n(1 - \sqrt{2/3}) \leq k \leq n\sqrt{2/3}$. Finally, the function $k \mapsto n!/\binom{n}{k}$, when restricted to some interval $[\![n(1-\alpha), \alpha n]\!]$, reaches its maximum at $k = \alpha n$, hence the upper bound $|r^{-1}(1)| \leq n!/\binom{n}{n\sqrt{2/3}}$. $\square$

References

1. Abío, I., Gange, G., Mayer-Eichberger, V., Stuckey, P.J.: On CNF encodings of decision diagrams. In: Quimper, C.-G. (ed.) CPAIOR 2016. LNCS, vol. 9676, pp. 1–17. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-33954-2_1

2. Abío, I., Nieuwenhuis, R., Oliveras, A., Rodríguez-Carbonell, E., Mayer-Eichberger, V.: A new look at BDDs for pseudo-boolean constraints. J. Artif. Intell. Res. **45**, 443–480 (2012). https://doi.org/10.1613/jair.3653

3. Aloul, F.A., Ramani, A., Markov, I.L., Sakallah, K.A.: Generic ILP versus specialized 0–1 ILP: an update. In: IEEE/ACM International Conference on Computer-aided Design, ICCAD, pp. 450–457 (2002). https://doi.org/10.1145/774572.774638

4. Bacchus, F.: GAC via unit propagation. In: Bessière, C. (ed.) CP 2007. LNCS, vol. 4741, pp. 133–147. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74970-7_12

5. Bailleux, O., Boufkhad, Y., Roussel, O.: New encodings of pseudo-boolean constraints into CNF. In: Kullmann, O. (ed.) SAT 2009. LNCS, vol. 5584, pp. 181–194. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02777-2_19

6. Bordeaux, L., Marques-Silva, J.: Knowledge compilation with empowerment. In: Bieliková, M., Friedrich, G., Gottlob, G., Katzenbeisser, S., Turán, G. (eds.) SOFSEM 2012. LNCS, vol. 7147, pp. 612–624. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27660-6_50

7. Boros, E., Hammer, P.L., Minoux, M., Rader, D.J.: Optimal cell flipping to minimize channel density in VLSI design and pseudo-boolean optimization. Discrete Appl. Math. **90**(1–3), 69–88 (1999). https://doi.org/10.1016/S0166-218X(98)00114-0

8. Bova, S., Capelli, F., Mengel, S., Slivovsky, F.: Knowledge compilation meets communication complexity. In: International Joint Conference on Artificial Intelligence, IJCAI, pp. 1008–1014 (2016). http://www.ijcai.org/Abstract/16/147

9. Bryant, R.E., Lahiri, S.K., Seshia, S.A.: Deciding CLU logic formulas via boolean and pseudo-boolean encodings. In: International Workshop on Constraints in Formal Verification, CFV (2002)

10. Darwiche, A., Marquis, P.: A knowledge compilation map. J. Artif. Intell. Res. **17**, 229–264 (2002). https://doi.org/10.1613/jair.989

11. Eén, N., Sörensson, N.: Translating pseudo-boolean constraints into SAT. J. Satisf. Boolean Model. Comput. **2**(1–4), 1–26 (2006). https://satassociation.org/jsat/index.php/jsat/article/view/18

12. Gange, G., Stuckey, P.J.: Explaining propagators for s-DNNF circuits. In: Beldiceanu, N., Jussien, N., Pinson, É. (eds.) CPAIOR 2012. LNCS, vol. 7298, pp. 195–210. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29828-8_13

13. Gopalan, P., Klivans, A.R., Meka, R., Štefankovič, D., Vempala, S.S., Vigoda, E.: An FPTAS for #Knapsack and related counting problems. In: IEEE Symposium on Foundations of Computer Science, FOCS, pp. 817–826 (2011). https://doi.org/10.1109/FOCS.2011.32

14. Hosaka, K., Takenaga, Y., Kaneda, T., Yajima, S.: Size of ordered binary decision diagrams representing threshold functions. Theor. Comput. Sci. **180**(1–2), 47–60 (1997). https://doi.org/10.1016/S0304-3975(97)83807-8

15. Ivănescu, P.L.: Some network flow problems solved with pseudo-boolean programming. Oper. Res. **13**(3), 388–399 (1965)

16. Jung, J.C., Barahona, P., Katsirelos, G., Walsh, T.: Two encodings of DNNF theories. In: ECAI Workshop on Inference Methods Based on Graphical Structures of Knowledge (2008)
17. Kučera, P., Savický, P.: Propagation complete encodings of smooth DNNF theories. CoRR abs/1909.06673 (2019). http://arxiv.org/abs/1909.06673
18. Le Berre, D., Marquis, P., Mengel, S., Wallon, R.: Pseudo-boolean constraints from a knowledge representation perspective. In: International Joint Conference on Artificial Intelligence, IJCAI, pp. 1891–1897 (2018). https://doi.org/10.24963/ijcai.2018/261
19. Marquis, P.: Compile! In: AAAI Conference on Artificial Intelligence, AAAI, pp. 4112–4118 (2015). http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9596
20. Mengel, S., Wallon, R.: Revisiting graph width measures for CNF-encodings. In: Janota, M., Lynce, I. (eds.) SAT 2019. LNCS, vol. 11628, pp. 222–238. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-24258-9_16
21. Papaioannou, S.G.: Optimal test generation in combinational networks by pseudo-boolean programming. IEEE Trans. Comput. **26**(6), 553–560 (1977). https://doi.org/10.1109/TC.1977.1674880
22. Wegener, I.: Branching programs and binary decision diagrams. SIAM (2000). http://ls2-www.cs.uni-dortmund.de/monographs/bdd/