Proof Complexity of Propositional Model Counting

Olaf Beyersdorff
Friedrich-Schiller-Universität Jena, Germany

Tim Hoffmann
Friedrich-Schiller-Universität Jena, Germany

Luc Nicolas Spachmann
Friedrich-Schiller-Universität Jena, Germany

Abstract

Recently, the proof system MICE for the model counting problem #SAT was introduced by Fichte, Hecher and Roland (SAT'22). As demonstrated by Fichte et al., the system MICE can be used for proof logging for state-of-the-art #SAT solvers.

We perform a proof-complexity study of MICE. For this we first simplify the rules of MICE and obtain a calculus MICE′ that is polynomially equivalent to MICE. Our main result establishes an exponential lower bound for the number of proof steps in MICE′ (and hence also in MICE) for a specific family of CNFs.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases model counting, #SAT, proof complexity, proof systems, lower bounds

Digital Object Identifier 10.4230/LIPIcs.SAT.2023.2

Funding *Olaf Beyersdorff*: Carl-Zeiss Foundation and DFG grant BE 4209/3-1.
Tim Hoffmann: Carl-Zeiss Foundation.

1 Introduction

The problem to decide whether a Boolean formula is satisfiable (SAT) is one of central problems in computer science, both theoretically and practically. From the theoretical side, SAT is the canonical NP-complete problem [14], making it intractable unless P=NP. From the practical side, the "SAT revolution" [31] with the evolution of practical SAT solvers has turned SAT into a tractable problem for many industrial instances [5].

In this paper we consider the *model counting problem* (#SAT) which asks how many satisfying assignments a given Boolean formula has. While #SAT is obviously a generalization of SAT, it is presumably much harder. #SAT is the canonical complete problem for the function class #P. While FP=#P would imply P=NP, it is known that FP=#P is even equivalent to P=PP. The power of #SAT is also illustrated by Toda's theorem [30] stating that any problem in the polynomial hierarchy can be solved in polynomial time with oracle access to #SAT.

Despite its higher complexity, #SAT solving has been actively pursued through the past two decades [20] and a number of #SAT solvers have been developed throughout the years. In fact, the past years have witnessed increased interest in #SAT solving with an annual model counting competition being organised since 2020 as part of the SAT conference [17]. #SAT solvers allow to tackle a large variety of real-world questions, including all kinds of problems in the areas probabilistic reasoning [2, 25], risk analysis [16, 34] and explainable artificial intelligence [3, 28].

Unlike in SAT solving where conflict-driven clause learning (CDCL) [26] dominates the scene, there are a number of conceptually different approaches to #SAT solving, including the lifting of standard techniques from SAT-solving [29], employing knowledge compilation [24], and via dynamic programming [19]. While some approaches try to approximate the number of solutions, we will only consider exact model counting in the following.

There is a tight correspondence between practical SAT solving and propositional proof systems [9]. While we know that in principle every SAT solver implicitly defines a proof system, a seminal result of [1, 27] established that CDCL (at least in its nondeterministic version) is equivalent to the resolution proof system. However, practical CDCL with e.g. the VSIDS heuristics corresponds to an exponentially weaker proof system than resolution [32]. In the same vein, there has recently been a line of research to understand the correspondence between solvers for quantified Boolean formulas (QBF) and QBF resolution proof systems [4, 6, 7].

This correspondence between solvers and proofs is not only of theoretical, but also of immense practical interest as it can be used for *proof logging*, i.e. for certifying the correctness of solvers on unsatisfiable SAT or QBF instances. Optimised proof systems have been devised in terms of RAT/DRAT for SAT [22, 33] and QRAT for QBF [23] for this purpose. These proof systems aim to capture all modern solving techniques, including preprocessing and therefore tend to be very powerful [10, 13]. In particular, in contrast to weak proof systems such as resolution, no lower bounds are known for RAT or QRAT.

In sharp contrast, far less is known about the correspondence of model counting solvers to proof systems. To our knowledge, there are currently two proof systems for #SAT. One is a static proof system based on decision DNNFs called kcps(#SAT) (the acronym stands for Knowledge Compilation based Proof System for #SAT) [11]. The other, a line based proof system called MICE [18] (the acronym stands for Model-counting Induction by Claim Extension), was just introduced at the last SAT conference [18]. Interestingly, the system MICE not only provides a theoretical proof system for #SAT, but also allows proof logging for a number of state-of-the-art solvers in model counting, including sharpSAT [29], DPDB [19] and D4 [24], as demonstrated in [18]. Hence MICE proofs can be used to verify the correctness of answers of these #SAT solvers.

1.1 Our Contributions

We perform a proof complexity analysis of the #SAT proof system MICE from [18]. Prior to this paper, no proof complexity results for MICE were known. Our results can be summarised as follows.

(a) A simplified proof system MICE'. We analyse the proof system MICE and define a somewhat simplified calculus MICE′. Lines in MICE are of the form $((F, V), A, c)$ where F is a propositional formula V is a set of variables, A is a partial assignment and $c \in \mathbb{N}$. Semantically, these lines express that the formula F under the partial assignment A has precisely c models. The system MICE then employs four rules to derive new lines with the ultimate goal to derive a line $((F, \mathsf{vars}(F)), \emptyset, c)$. Thus in the ultimate line, c is the number of models of the formula F.

The four rules of the system include one axiom rule for satisfying total assignments and three rules to compose, join and extend existing lines. All the rules have a rather extensive set of side conditions to verify their applicability. For the composition rule this even includes an external resolution proof to check that the composition of claims in the rule indeed covers all models.

The variable set V does not feature in the semantical explanation above. While it might be tempting to choose $V = \mathsf{vars}(F)$ for all lines (as is done in the final claim), we show that this restriction is too strong and results in an exponentially weaker system. Nevertheless, we show that we can slightly adapt the rules of MICE (in particular the extension rule) and obtain a system MICE′ for which we can impose $V = \mathsf{vars}(F)$ for all lines without weakening the system. Lines in MICE′ therefore can take the form (F, A, c). This allows

allows to eliminate and simplify some of the side conditions for the original rules of MICE when transferring to MICE′. Our simplified system MICE′ is as strong as MICE in terms of simulations (Propositions 16 and 17). Hence also MICE′ can be used for proof logging for the #SAT solvers mentioned above.

(b) Lower bounds for MICE and MICE′. In our main result we show an exponential lower bound for the proof size in MICE′ (and hence also for MICE) for a specific family of CNFs.

As mentioned above, the composition rule of MICE (and MICE′) incorporates resolution proofs. Exploiting this feature, it is not too hard to transfer resolution lower bounds to MICE′. In fact, we can show that on unsatisfiable formulas, resolution is polynomially equivalent to MICE′ (Theorem 18).

However, we would view such a transferred resolution lower bound not as a "genuine" and interesting lower bound for MICE′. We therefore show a stronger bound for MICE′ for *the number of proof steps* (where we disregard the size of the attached resolution proofs). In our main result we show a lower bound of $2^{\Omega(n)}$ for the number of proof steps for a specific set of CNFs, termed XOR-PAIRS$_n$, based on the parity function (Theorem 23). Technically, our lower bound is established by showing that in MICE′ proofs of XOR-PAIRS$_n$, all applications of the join and extension rules preserve the model count.

1.2 Relations to DNNFs

One of the anonymous reviewers highlighted that there is a close connection between our work here and Decomposable Negation Normal Forms (DNNFs) as investigated in [8, 11, 12]. We were not aware of that work and would like to thank the reviewer for pointing that out.

In particular, it appears that from a MICE′ proof a decision DNNF can be efficiently extracted. Hence, alternatively to our directly obtained lower bound for MICE′ in Section 5, one could employ decision DNNF lower bounds as shown via communication complexity in [8] for MICE′ lower bounds.

1.3 Organisation

The remainder of this paper is organised as follows. After reviewing some standard notions from propositional logic and proof systems in Section 2, we revise the #SAT proof system MICE from [18] in Section 3 and show some properties of the system. This gives rise to a simplified proof system MICE′ which we define in Section 4. Section 5 contains our main results on the exponential lower bound for MICE′ (and hence for MICE). We conclude in Section 6 with relations to some open questions and future directions.

2 Preliminaries

We introduce some notations used in this paper. A literal l is a variable z or its negation $\overline{z}$, with $\mathsf{var}(l) = z$. A clause is a disjunction of literals, a conjunctive normal form (CNF) is a conjunction of clauses. Often, we write clauses as sets of literals and formulas as sets of clauses. We assume that every propositional formula is written in CNF.

For a formula F, $\mathsf{vars}(F)$ denotes the set of all variables that occur in F, and $\mathsf{lits}(F)$ is the set of all literals of F. If $C \in F$ is a clause and $V \subseteq \mathsf{vars}(F)$ is a set of variables, we define $C|_V = \{l \in C \mid \mathsf{vars}(l) \in V\}$ and $F|_V$ denotes the formula F with every clause C replaced by $C|_V$. An assignment is a function α mapping variables to Boolean values. If a function F evaluates to true under an assignment α, we say α satisfies F and write $\alpha \models F$. We also

allow α to be a partial assignment to $\mathsf{vars}(F)$ or to contain variables not occurring in F. Occasionally, we interpret an assignment as a CNF consisting of precisely those unit clauses that specify the assignment. Therefore, the set operations are well defined for formulas and assignments. We say that two assignments are consistent if their union is satisfiable. For some set of variables X, $\langle X \rangle$ denotes the set of all $2^{|X|}$ possible assignments to X.

In this paper we are interested in proof systems as introduced in [15]. Formally, a proof system for a language L is a polynomial-time computable function f with $\mathsf{rng}(f) = L$. If $f(w) = x$, then w is called f-proof of $x \in L$. In order to compare proof systems we need the notion of simulations. Let f and g be proof systems for language L. We say that f simulates g, if for any g-proof w there exists an f-proof w' with $|w'| = |w|^{O(1)}$ and $f(w') = g(w)$. If we can compute w' in polynomial time from w, we say that f p-simulates g. Two proof systems are (p-)equivalent if they (p-)simulate each other.

For the language UNSAT of unsatisfiable CNFs, resolution is arguably the most studied proof system. It operates on Boolean formulas in CNF and has only one rule. This resolution rule can derive $C \cup D$ from $C \cup \{x\}$ and $D \cup \{\overline{x}\}$ with arbitrary clauses C, D and variable x. A resolution refutation of a CNF is a derivation of the empty clause $\square$. We sometimes add a weakening rule that enables us to derive $C \cup D$ from C for arbitrary clauses C and D. However, it is well-known that any resolution refutation that uses weakening can be efficiently transformed into a resolution refutation without weakening.

3 The Proof System MICE for #SAT

In this section we recall the MICE proof system for #SAT from [18] and show some basic properties of the system.

▶ **Definition 1** ([18]). *A* claim *is a triple* $((F, V), A, c)$ *where* F *is a propositional formula in CNF,* V *is a set of variables,* A *is an assignment with* $\mathsf{vars}(A) \subseteq V$ *and* $c \in \mathbb{N}$. *For such a claim, let* $\mathsf{Mod}_A(F, V) := \{\alpha \in \langle V \rangle \mid \alpha \models F \cup A\}$. *The claim is* correct *if* $c = |\mathsf{Mod}_A(F, V)|$.

Claims will be the lines in our proof systems for model counting. Semantically, they describe that the formula F under the partial assignment A has exactly c models. The partial assignment A is sometimes also referred to as the assumption. What is perhaps a bit mysterious at this point is the role of the variable set V. We will get to this shortly.

The rules of MICE are Exactly One Model (1-Mod), Composition (Comp), Join (Join) and Extension (Ext). They are specified in Figure 1. We give some intuition on the rules. The axiom rule (1-Mod) states that if a complete assignment A satisfies a formula F, then F has exactly one model under A.

With (Comp) we can sum up model counts of a formula F under different partial assignments $A_1, \ldots, A_n$ in order to weaken the assumption to a partial assignment A. This is only sound if the solutions of F under assumptions $A_1, \ldots, A_n$ form a disjoint partition of the full solution space of F under A. That this is indeed the case can be verified with an independent proof, e.g. in propositional resolution. This proof is called an *absence of models statement*.

The (Join) rule allows us to multiply the model counts of two formulas that are completely independent restricted to the assumptions. Finally, with (Ext), we can extend simultaneously all models, i.e. we enlarge the formula and assumption without changing the count.

We can now formally define MICE proofs.

▶ **Definition 2** (Fichte, Hecher, Roland [18]). *A* MICE trace *is a sequence* $\pi = (I_1, \ldots, I_k)$ *where for each* $i \in [k]$, *either*

- *I_i is a claim if I_i is derived by one of (1-Mod), (Join), (Ext) or*
- *$I_i = (I, \rho)$ if the claim I is derived by (Comp) and ρ is the resolution refutation for the respective absence of models statement.*

A MICE proof *of a formula φ is a* MICE *trace $\pi = (I_1, \dots, I_k)$ where I_k is (or contains in case of (Comp)) the claim $((\varphi, \mathsf{vars}(\varphi)), \emptyset, c)$ for some $c \in \mathbb{N}$.*

In [18] it is shown that MICE is a sound and complete proof system for #SAT.

For measuring the *proof size*, we use two natural options. $s(\pi)$ notates the size of π which is the total number of claims plus the number of clauses in resolution proofs in the absence of models statements. $c(\pi)$ counts only the number of claims a proof has which is exactly the number of inference steps that the proof needs.

Exactly One Model.

$$\frac{}{((F,V),A,1)} \qquad \text{(1-Mod)}$$

- (O-1) $\mathsf{vars}(A) = V$,
- (O-2) A satisfies F.

Composition.

$$\frac{((F,V),A_1,c_1),\dots,((F,V),A_n,c_n)}{((F,V),A,\sum_{i\in[n]} c_i)} \qquad \text{(Comp)}$$

- (C-1) $\mathsf{vars}(A_1) = \mathsf{vars}(A_2) = \cdots = \mathsf{vars}(A_n)$ and $A_i \neq A_j$ for $i \neq j$,
- (C-2) $A \subseteq A_i$ for all $i \in [n]$,
- (C-3) there exists a resolution refutation of $A \cup \{C|_V \mid C \in F\} \cup \{\overline{A}_i \mid i \in [n]\}$. Such a refutation is included into the trace and is called an *absence of models statement*.

Join.

$$\frac{((F_1,V_1),A_1,c_1),((F_2,V_2),A_2,c_2)}{((F_1\cup F_2, V_1\cup V_2), A_1\cup A_2, c_1\cdot c_2)} \qquad \text{(Join)}$$

- (J-1) A_1 and A_2 are consistent,
- (J-2) $V_1 \cap V_2 \subseteq \mathsf{vars}(A_i)$ for $i \in \{1,2\}$,
- (J-3) $\mathsf{vars}(F_i) \cap ((V_1 \cup V_2) \setminus V_i) = \emptyset$ for $i \in \{1,2\}$.

Extension.

$$\frac{((F_1,V_1),A_1,c)}{((F,V),A,c)} \qquad \text{(Ext)}$$

- (E-1) $F_1 \subseteq F$, $V_1 \subseteq V$,
- (E-2) $V \setminus V_1 \subseteq \mathsf{vars}(A)$,
- (E-3) $A|_{V_1} = A_1$,
- (E-4) A satisfies $F \setminus F_1$,
- (E-5) for every $C \in F_1$: $A|_{V\setminus V_1}$ does not satisfy C.

Figure 1 Inference rules for MICE [18].

In a correct claim $((F,V),A,c)$ the count c is uniquely determined by the the formula F, set of variables V and assumption A. Therefore, we often omit c and refer to the claim as $((F,V),A)$. To ease notation we will usually just write a MICE proof as as sequence of

claims $I_1, \ldots, I_m$ and do not explicitly record the used absence of models statements. We just assume that whenever we use (Comp), the necessary resolution refutation is part of the MICE proof.

If a formula F is satisfied by the partial assignment A, we can set the remaining variables arbitrarily. Therefore, the component $(F, \mathsf{vars}(F))$ has exactly $2^{|\mathsf{vars}(F)|-|\mathsf{vars}(A)|}$ models under assumption A. The following construction shows that we can efficiently derive the corresponding claim in MICE.

▶ **Proposition 3.** *If some assumption A satisfies an arbitrary formula F, there is a* MICE *derivation of the claim $I = ((F, \mathsf{vars}(F)), A, 2^{|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|})$ with $s(\pi) = 7\cdot(|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|)$ and $c(\pi) = 4 \cdot (|\mathsf{vars}(F) \setminus \mathsf{vars}(A)|)$.*

Proof. Let $\mathsf{vars}(F) \setminus \mathsf{vars}(A) = \{x_1, \ldots, x_n\}$. For every $i \in [n]$ we derive $I_i^1 = ((\emptyset, \mathsf{vars}(A) \cup \{x_i\}), A \cup \{x_i\}, 1)$ and $I_i^0 = ((\emptyset, \mathsf{vars}(A) \cup \{x_i\}), A \cup \{\overline{x}_i\}, 1)$ with (1-Mod). This is possible since every assignment satisfies the empty formula. With (Comp) we get $I_i = ((\emptyset, \mathsf{vars}(A) \cup \{x_i\}), A, 2)$ using the absence of models statement $\rho_i = ((x_i), (\overline{x}_i), \square)$. We use (Join) of I_1 and I_2, then (Join) of the result and I_3, and so on. The requirements (J-1), (J-2), and (J-3) are satisfied. In this way we get $((\emptyset, \mathsf{vars}(F)), A, 2^{|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|})$. We use (Ext) to obtain $I = ((F, \mathsf{vars}(F)), A, 2^{|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|})$. It is easy to see that all requirements (E-1) to (E-5) are satisfied. For (E-4), we use that A satisfies F. In total we use $4n$ MICE steps to derive I and we have n absence of models statements with 3 clauses each. ◀

We investigate some properties that any claim in a MICE proof has to fulfill. We assume that any MICE proof has no redundant claims, i.e. in the corresponding proof dag, there is a path from any node to the final claim. We also observe that for all inference rules, the derived F and V never shrink. This leads to the following two observations:

▶ **Observation 4.** *If $((F, V), A)$ is derived from $((F_1, V_1), A_1)$ in a* MICE *trace (not necessarily in one step), then $F_1 \subseteq F$ and $V_1 \subseteq V$.*

Therefore, any claim $((F, V), A)$ in a MICE *proof of φ fulfills $F \subseteq \varphi$ and $V \subseteq \mathsf{vars}(\varphi)$.*

From Definition 1 it is not obvious how F and V are related. Intuitively, one might be tempted to set $V = \mathsf{vars}(F)$ for any claim $((F, V), A)$. However, this would make the proof system exponentially weaker as we will see later. Lemma 6 will show that we can at least assume $\mathsf{vars}(F) \subseteq V$ for every claim. To show this we need the following lemma:

▶ **Lemma 5.** *For any claim $((F, V), A)$ and any variable x, if $x \in \mathsf{vars}(F) \setminus V$, then literals x and $\overline{x}$ cannot both occur in F.*

Proof Sketch. Suppose there exists such an x. Since $((F, V), A)$ is not redundant, there is a path to the final claim. Thus, there have to be claims $((F_1, V_1), A_1)$ and $((F_2, V_2), A_2)$ directly adjacent in the path with $F \subseteq F_1 \subseteq F_2$, $V \subseteq V_1 \subseteq V_2$ and $x \notin V_1$, $x \in V_2$. Now $((F_2, V_2), A_2)$ is directly derived from $((F_1, V_1), A_1)$ in one step. We can argue that this is not possible. ◀

▶ **Lemma 6.** *Let a formula φ and a* MICE *proof π for φ be given. Then there is a* MICE *proof π' satisfying $\mathsf{vars}(F) \subseteq V$ for any claim $((F, V), A) \in \pi'$ such that $s(\pi') = O(s(\pi)^3)$ and $c(\pi') = c(\pi)$.*

Proof Sketch. Let $\pi = (I_1, \ldots, I_m)$ with $I_i = ((F_i, V_i), A_i)$. Because of Lemma 5, for any $i \in [m]$, we can assume that there is no variable $x \in \mathsf{vars}(F_i) \setminus V_i$ that occurs in both polarities in F_i. Let $\alpha_i \in \langle \mathsf{vars}(F_i) \setminus V_i \rangle$ be the assignment that does not satisfy any clause in F_i, i.e. if x is in F_i we assign $\alpha_i(x) = 0$ and vice versa. For every claim I_i, α_i exists and it is unique. We define

$f(((F_i, V_i), A_i)) := ((F_i, V_i \cup \mathsf{vars}(F_i)), A_i \cup \alpha_i)$

with the unique α_i defined above. We show by induction that $(f(I_1), \ldots, f(I_m))$ is a valid MICE proof for φ. ◀

In the following we always assume $\mathsf{vars}(F) \subseteq V$ for any claim $((F, V), A)$. With this requirement, the conditions of the inference rules can be simplified.

▶ **Corollary 7.** *If we require* $\mathsf{vars}(F) \subseteq V$ *for every claim* $((F, V), A)$*, the following simplifications for the* MICE *rules apply:*

- *We can simplify the absence of models statement in the requirement (C-2) to be a refutation of* $F \cup A \cup \{\overline{A_i} \mid i \in [n]\}$*.*
- *We can remove condition (J-3) for (Join).*
- *We can remove condition (E-5) for (Ext).*

However, imposing the stronger condition $\mathsf{vars}(F) = V$ for every claim $((F, V), A)$ would make the proof system exponentially weaker as we illustrate with the next proposition.

▶ **Lemma 8.** *There is a family of formulas* $(T_n)_{n \in \mathbb{N}}$ *such that for both measures* $s(\cdot)$ *and* $c(\cdot)$ *holds:*

- T_n *has polynomial-size* MICE *proofs and*
- *if* $\mathsf{vars}(F) = V$ *is required for all claims* $((F, V), A)$*, the shortest* MICE *proof of* T_n *has exponential size.*

Proof Sketch. Consider the formula T_n that only has one clause $(x_1 \vee x_2 \vee \cdots \vee x_n)$.

To construct a polynomial-size MICE proof, we derive $((\emptyset, \mathsf{vars}(T_n)), \{x_1 = 1\}, 2^{n-1})$ with a small number of applications of (1-Mod) and (Join). We get $((T_n, \mathsf{vars}(T_n)), \{x_1 = 1\}, 2^{n-1})$ with (Ext). Similarly, we derive $((T_n, \mathsf{vars}(T_n)), \{x_1 = 0, x_2 = 1\}, 2^{n-2})$ and so on. With applications of (Comp) we combine these claims to $((T_n, \mathsf{vars}(T_n)), \emptyset, 2^n - 1)$.

We can show that any MICE proof with the additional requirement $\mathsf{vars}(F) = V$ needs to have a claim $((T_n, \mathsf{vars}(T_n)), \alpha)$ for every model $\alpha \in \mathsf{Mod}(T_n)$. Since T_n has $2^n - 1$ models, the proof has size $2^{\Omega(n)}$. ◀

4 A Simplified Proof System MICE' for #SAT

We now adapt MICE to a new proof system MICE′ that is as strong as MICE and only uses claims $((F, V), A)$ with components satisfying $V = \mathsf{vars}(F)$. Therefore, we can drop the explicit mentioning of the variable set V and only need to specify the formula F. This makes the resulting proof system more intuitive and easier to investigate for lower bounds.

The rules of MICE′ are Axiom (Ax), Composition (Comp'), Join (Join') and Extension (Ext'). They are specified in Figure 2.

The intuition for the rules (Comp') and (Join') are very similar to (Comp) and (Join) from MICE. The (Ax) rule enables us to derive the claim $(\emptyset, \emptyset, 1)$ which is trivially true. (Ext') is also similar to (Ext) with one important difference: If we use (Ext) in MICE, the assumption has to assign all variables that are added to the claim. As result, we extend one model of the original claim to one new model. In (Ext') however, this is not necessarily the case. As long as the new assumption satisfies all added clauses, we are allowed to leave new introduced variables unassigned in the assumption. Like this we extend every model of the original claim to a set of new models, one for every possible assignment of these unassigned variables.

Axiom.

$$\frac{}{(\emptyset, \emptyset, 1)} \qquad (\mathsf{Ax})$$

Composition.

$$\frac{(F, A_1, c_1), \ldots, (F, A_n, c_n)}{(F, A, \sum_{i \in [n]} c_i)} \qquad (\mathsf{Comp'})$$

- (C-1) $\mathsf{vars}(A_1) = \mathsf{vars}(A_2) = \cdots = \mathsf{vars}(A_n)$ and $A_i \neq A_j$ for $i \neq j$,
- (C-2) $A \subseteq A_i$ for all $i \in [n]$,
- (C-3) there exists a resolution refutation of $A \cup F \cup \{\overline{A}_i \mid i \in [n]\}$. Such a refutation is included into the trace and is called an *absence of models statement*.

Join.

$$\frac{(F_1, A_1, c_1), (F_2, A_2, c_2)}{(F_1 \cup F_2, A_1 \cup A_2, c_1 \cdot c_2)} \qquad (\mathsf{Join'})$$

- (J-1) A_1 and A_2 are consistent,
- (J-2) $\mathsf{vars}(F_1) \cap \mathsf{vars}(F_2) \subseteq \mathsf{vars}(A_i)$ for $i \in \{1, 2\}$.

Extension.

$$\frac{(F_1, A_1, c_1)}{(F, A, c_1 \cdot 2^{|\mathsf{vars}(F) \setminus (\mathsf{vars}(F_1) \cup \mathsf{vars}(A))|})} \qquad (\mathsf{Ext'})$$

- (E-1) $F_1 \subseteq F$,
- (E-2) $A|_{\mathsf{vars}(F_1)} = A_1$,
- (E-3) A satisfies $F \setminus F_1$.

Figure 2 Inference rules for MICE′.

▶ **Definition 9** (Adapted Proof System MICE′). *A* claim *is a triple* (F, A, c) *with* $\mathsf{vars}(A) \subseteq \mathsf{vars}(F)$. *For such a claim, let* $\mathsf{Mod}_A(F) := \{\alpha \in \langle \mathsf{vars}(F) \rangle \mid \alpha \models F \cup A\}$. *The claim is* correct *if* $c = |\mathsf{Mod}_A(F)|$. *The rules of* MICE′ *are (Ax), (Comp'), (Join') and (Ext'). The notions of* MICE′ traces *and* MICE′ proofs *are defined analogously as for* MICE. *Furthermore, we use the same two measures for the proof size* $s(\cdot)$ *and* $c(\cdot)$.

As in the MICE proof system we often omit the count c of claims and assume that no redundant claims exist in MICE′ proofs, i.e. all claims are connected to the final claim.

We prove that all four derivation rules are sound, i.e. for every derived claim (F, A, c) holds $c = |\mathsf{Mod}_A(F)|$. In doing so, we will also provide some intuition on the semantic meaning of the rules.

▶ **Lemma 10.** *The inference rules of* MICE′ *are sound.*

Proof Sketch. To prove the soundness of every MICE′ rule, we associate every claim (F, A, c) with the set containing exactly the c models in $\mathsf{Mod}_A(F)$. With this interpretation, we can specify how every rule modifies these models. This way, we can show that the resulting model count is indeed correct for every MICE′ rule.

The soundness of (Ax) is obvious, since $\mathsf{Mod}_\emptyset(\emptyset) = \{\emptyset\}$.

To show soundness of (Comp'), let $(F, A, \sum_{i\in[n]} c_i)$ be derived with (Comp') from correct claims $(F, A_1, c_1), \ldots, (F, A_n, c_n)$. Then we can show

$$\mathsf{Mod}_A(F) = \{\alpha \in \langle \mathsf{vars}(F)\rangle \mid \alpha \models F \cup A\}.$$

Next, we show soundness of (Join'). For this, let $(F_1 \cup F_2, A_1 \cup A_2, c_1 \cdot c_2)$ be derived with (Join') from correct claims (F_1, A_1, c_1) and (F_2, A_2, c_2). We can show that

$$\mathsf{Mod}_{A_1\cup A_2}(F_1 \cup F_2) = \{\alpha_1 \cup \alpha_2 \mid \alpha_1 \in \mathsf{Mod}_{A_1}(F_1), \alpha_2 \in \mathsf{Mod}_{A_2}(F_2)\}.$$

Finally we have to show that (Ext') is sound. Assume (F, A, c) is derived with (Ext') from the correct claim (F_1, A_1, c_1). We can show

$$\mathsf{Mod}_A(F) = \{\alpha \cup (A \setminus A_1) \cup \beta \mid \alpha \in \mathsf{Mod}_{A_1}(F_1), \beta \in \langle \mathsf{vars}(F) \setminus (\mathsf{vars}(F_1) \cup \mathsf{vars}(A))\rangle\}.$$

Therefore, claims derived with MICE′ are correct. ◀

▶ **Corollary 11.** *Let claim $I = (F, A)$ and a model $\alpha \in \mathsf{Mod}_A(F)$ be given.*

- *If I is derived with (Comp') using claims $(F, A_1), \ldots, (F, A_n)$, then there exists exactly one $i \in [n]$ such that $\alpha \in \mathsf{Mod}_{A_i}(F_i)$.*
- *If I is derived with (Join') using claims (F_1, A_1) and (F_2, A_2), then for both $i \in [2]$ we have $\alpha|_{\mathsf{vars}(F_i)} \in \mathsf{Mod}_{A_i}(F_i)$.*
- *If I is derived with (Ext') using claim (F_1, A_1), then $\alpha|_{\mathsf{vars}(F_1)} \in \mathsf{Mod}_{A_1}(F_1)$.*

We introduce an additional rule (SA) which is similar to the construction in Proposition 3.

▶ **Definition 12** (Satisfying Assumption Rule). *Under the condition (S-1): A satisfies F, we allow to derive*

$$\frac{}{(F, A, 2^{|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|})} \quad (SA).$$

This rule is sound and does not make MICE′ proofs much shorter.

▶ **Lemma 13.** *(SA) is sound. Further, if formula φ has a MICE′ proof π that can use the additional rule (SA), then there exists a MICE′ proof π' of φ with $s(\pi') = s(\pi) + 1$ and $c(\pi') = c(\pi) + 1$.*

Proof. Assume that we applied (SA) in π to derive claim $I = (F, A, 2^{|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|})$. Then we can derive I without (SA) with two MICE′ steps in the following way. We use (Ax) to get $(\emptyset, \emptyset, 1)$ and then (Ext') to derive I. It is easy to see that conditions (E-1) and (E-2) are fulfilled. (E-3) follows directly from (S-1). The resulting counts are the same since $1 \cdot 2^{|\mathsf{vars}(F)\setminus(\mathsf{vars}(F_1))\cup\mathsf{vars}(A))|} = 2^{|\mathsf{vars}(F)\setminus\mathsf{vars}(A)|}$. Since we can simulate (SA) with the other sound MICE′ rules, (SA) is sound as well. If we replace all applications of (SA) like this, then the proof size increases at most by one, as we need (Ax) only once in the proof. ◀

To justify our definition of MICE′ we have to show that it is indeed a proof system for #SAT.

▶ **Theorem 14.** MICE′ *is a sound and complete proof system for #SAT.*

Proof. The soundness of MICE′ follows directly from the soundness of the inference rules as shown in Lemma 10.

Next, we show that MICE′ is complete. For this, let an arbitrary formula φ be given. We can derive $I_\alpha = (\varphi, \alpha, 1)$ for every $\alpha \in \mathsf{Mod}(\varphi)$ with (SA). For all these models together there is an absence of models statement. Therefore, we can derive $(\varphi, \emptyset, |\mathsf{Mod}(\varphi)|)$ with (Comp') from all claims I_α. Note that for unsatisfiable formulas we can derive the final claim with a single application of (Comp').

In proof systems, it is also necessary that proofs can be verified in polynomial time. This is possible in MICE′ since all conditions (C-1), (C-2), (C-3), (J-1), (J-2), (E-1), (E-2) and (E-3) are easy to check in polynomial time. ◀

Next, we show some basic properties of MICE′.

▶ **Lemma 15.** *Let claim (F_1, A_1) be used to derive (F, A) (not necessarily in one step). Then*

- $F_1 \subseteq F$,
- *if $x \in \mathsf{vars}(F_1) \cap \mathsf{vars}(A)$, then $x \in \mathsf{vars}(A_1)$ and $A(x) = A_1(x)$.*

Proof. Because every MICE′ rule does not decrease the formula F, the first property is obvious.

Let $((F_1, A_1), \ldots, (F_n, A_n) = (F, A))$ be a path in this derivation. It is easy to check that for all four inference rules of MICE′ we have $A_{i+1}|_{\mathsf{vars}(F_i)} \subseteq A_i$ for $i \in [n-1]$. We can restrict both sides and get

$$(A_{i+1}|_{\mathsf{vars}(F_i)})|_{\mathsf{vars}(F_1)} = A_{i+1}|_{\mathsf{vars}(F_i) \cap \mathsf{vars}(F_1)} = A_{i+1}|_{\mathsf{vars}(F_1)} \subseteq A_i|_{\mathsf{vars}(F_1)}.$$

Therefore,

$$A|_{\mathsf{vars}(F_1)} = A_n|_{\mathsf{vars}(F_1)} \subseteq A_{n-1}|_{\mathsf{vars}(F_1)} \subseteq \cdots \subseteq A_1|_{\mathsf{vars}}(F_1) = A_1.$$

From $A|_{\mathsf{vars}(F_1)} \subseteq A_1$ the second property follows. ◀

Using these properties, we can show that the new proof system MICE′ is polynomially equivalent to MICE. Note that this result is true for both measures of proof size $s(\cdot)$ and $c(\cdot)$. To prove this equivalence, we show both simulations separately.

First we show that MICE′ is at least as strong as MICE. This simulation is the more important one for this paper as it implies that lower bounds for MICE′ do also apply for MICE.

▶ **Proposition 16.** MICE′ *p-simulates* MICE.

Proof Sketch. Let $\pi = (I_1, \ldots, I_m)$ be a MICE proof of a given formula φ. We assume that $\mathsf{vars}(F) \subseteq V$ for all claims $((F, V), A)$ in π which is justified by Lemma 6. We can show by induction that for $f(((F, V), A)) := (F, A|_{\mathsf{vars}(F)})$ the sequence $\pi' = (f(I_1), \ldots, f(I_m))$ is a correct MICE′ proof of φ. ◀

Next we show that MICE′ is not stronger than MICE. Although this result is not needed for the lower bounds, it is nice to know how our new proof system MICE′ relates to MICE exactly.

▶ **Proposition 17.** MICE *p-simulates* MICE′.

Proof Sketch. Let $\pi = I_1, \ldots, I_n$ with $I_i = (F_i, A_i)$ be a MICE′ proof of a given formula φ. We define $f(I_i) := ((F_i, \mathsf{vars}(F_i)), A_i)$ and show that we can derive $f(I_k)$ using $f(I_1), \ldots, f(I_{k-1})$ with $O(|\mathsf{vars}(\varphi)|)$ MICE steps. ◀

5 Lower Bounds for MICE and MICE'

In this section we investigate the proof complexity of MICE′. For the analysis we use the two different measures of proof size.

First, we consider the proof size $s(\cdot)$. For that, we can easily lift known lower bounds from propositional resolution and get families of formulas that require MICE′ proofs of exponential size.

However, one could argue, that this is not the kind of hardness we are interested in. In the second part we get a stronger result by showing a lower bound for the number of inference steps $c(\cdot)$, i.e. we ignore the sizes of the absence of models statements.

5.1 Lower Bounds for the Proof Size

In this subsection we only consider the proof size $s(\cdot)$ that counts the number of claims plus the length of all resolution refutations. If we use MICE′ on unsatisfiable formulas, we have to prove that the formula has zero models. Hence, we can use MICE′ as proof system for the language UNSAT as well. We show that MICE′ is precisely as strong as resolution for unsatisfiable formulas.

▶ **Theorem 18.** MICE′ *is polynomially equivalent to* Res *for unsatisfiable formulas.*

Proof Sketch. Let φ be an arbitrary unsatisfiable formula.

We first show that Res is simulated by MICE′. Suppose π_{Res} is a resolution refutation of φ, then we can use π_{Res} as an absence of models statement and derive the final claim $(\varphi, \emptyset, 0)$ with a single application of (Comp) of zero claims.

Next, we show that MICE′ is simulated by Res. Let a MICE′ refutation $\pi = (I_1, \ldots, I_m)$ for φ be given with $I_i = (F_i, A_i, c_i)$. Further, let $\pi_{\mathsf{Res}} = (\varphi, X_1, X_2, \ldots, X_m)$ where X_i is a sequence of clauses defined as

$$X_i := \begin{cases} \text{empty sequence} & \text{if } c_i \neq 0 \\ (\overline{A}_i) & \text{if } I_i \text{ is derived by (Join') or (Ext')} \\ (C \cup \overline{A}_i \mid C \in \rho) & \text{if } I_i \text{ is derived by (Comp') and absence of models statement } \rho. \end{cases}$$

We can show that π_{Res} is a valid resolution trace (with weakening steps). ◀

Many hard families of formulas for resolution are known. One famous example is the pigeonhole formula family PHP for which an exponential lower bound for resolution was first shown in [21]. With Theorem 18 we can conclude that these hard formulas for resolution are also hard for MICE′.

▶ **Corollary 19.** *Any* MICE′ *proof* π *of* PHP_n *has size* $s(\pi) = 2^{\Omega(n)}$.

We note that it is also quite straightforward to obtain exponential proof size lower bounds for satisfiable formulas in MICE′ by forcing the system to refute some exponentially hard CNFs in absence of models statements.

5.2 Lower Bounds for the Number of Inference Steps

One could argue that unsatisfiable formulas such as PHP are not particularly interesting for model counting. We also note that they have very simple MICE′ proofs of just one step (as in the simulation of resolution by MICE′ in Theorem 18) and that their hardness for MICE′

stems solely from the fact that they are hard for resolution (and such resolution proofs need to be included as an absence of models statement). However, we would argue that this does not tell us much on the complexity of MICE′ proofs.

We therefore now tighten our complexity measure and consider the proof size measure $c(\cdot)$ that only counts the number of MICE′ inference steps which is exactly the number of claims a proof π has. This measure disregards the size of the accompanying resolution refutations and hence formulas such as PHP become easy.

In our main result we present a family of formulas that is exponentially hard with respect to this sharper measure of counting inference steps. Such hard formulas need to have many models as the following upper bound shows.

▶ **Observation 20.** *Every formula φ has a* MICE′ *proof π with $c(\pi) = |\mathsf{Mod}(\varphi)| + 2$.*

Proof. The MICE′ proof that we used to show the completeness in Theorem 14 needs one (Ax) step, $|\mathsf{Mod}(\varphi)|$ applications of (Ext'), and one application of (Comp'). ◀

Therefore, to show exponential lower bounds to the number of steps we will need formulas with $2^{\Omega(n)}$ models. Next, we show that MICE′ proofs for such formulas do not require claims with $c = 0$. In particular, we can assume that there are no such claims in the proofs.

▶ **Lemma 21.** *Let $\varphi \in SAT$ and π be a* MICE′ *proof of φ. Then there is a* MICE′ *proof π' of φ that has no claim with count $c = 0$ such that $s(\pi') = O(s(\pi)^2)$ and $c(\pi') \leq c(\pi)$.*

Proof Sketch. We consider an arbitrary claim I in the π with $c = 0$. Since I is not redundant, there is a path to the final claim. The final claim has count $c > 0$, since φ is satisfiable. Therefore, in this path there are two adjacent claims (F_1, A_1, c_1) and (F_2, A_2, c_2) with $c_1 = 0$ and $c_2 > 0$. We can argue that (F_2, A_2, c_2) is derived with (Comp'). We can adapt the absence of models statement such that (F_1, A_1, c_1) is not needed for this (Comp') application. ◀

Next, we introduce the family of formulas $(\mathsf{XOR\text{-}PAIRS}_n)_{n\in\mathbb{N}}$. They consist of variables x_i and z_{ij} for $i, j \in [n]$ and are satisfied exactly if $(z_{ij} = x_i \oplus x_j)$ for every pair $i, j \in [n]$.

▶ **Definition 22.** *The formula* $\mathsf{XOR\text{-}PAIRS}_n$ *consists of the clauses*

$$C^1_{ij} = (x_i \vee x_j \vee \overline{z}_{ij}),\ C^2_{ij} = (\overline{x}_i \vee x_j \vee z_{ij}),\ C^3_{ij} = (x_i \vee \overline{x}_j \vee z_{ij}),\ C^4_{ij} = (\overline{x}_i \vee \overline{x}_j \vee \overline{z}_{ij})$$

for $i, j \in [n]$.

▶ **Theorem 23.** *Any* MICE′ *proof π of* $\mathsf{XOR\text{-}PAIRS}_n$ *requires size $c(\pi) = 2^{\Omega(n)}$.*

We start with some observations and lemmas and prove the lower bound at the end of this section.

The *idea of the proof* is the following: The final claim has a large count. In order to get a large count with a small number of MICE′ steps, we have to use (Ext') or (Join') such that the previous counts get multiplied. However, we show that one factor of any such multiplication is always 1. As a result, the only way to increase the count is with (Comp'). We start with applications of (Ax) with count 1 and can only sum up those counts with (Comp'). As a result, we need an exponential number of summands.

▶ **Observation 24.** $\mathsf{XOR\text{-}PAIRS}_n$ *has 2^n models.*

Proof. We can set x_i arbitrarily for all $i \in [n]$ and have a unique assignment for the remaining z variables to satisfy $\mathsf{XOR\text{-}PAIRS}_n$. ◀

For the following arguments we will only consider MICE′ proofs of XOR-PAIRS$_n$ without redundant claims (i.e. all claims are connected to the final claim) and without claims with $c = 0$ (this is possible by Lemma 21). Our next lemma states that if we have some clause C_{ij} in a claim, then all missing clauses C_{ij} have to be satisfied by the assumption.

▶ **Lemma 25.** *Let (F, A) be an arbitrary claim in a* MICE′ *proof of* XOR-PAIRS$_n$*. If there are $i, j \in [n]$ such that $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F)$, then A has to satisfy every clause C_{ij}^k for $k \in [4]$ that is not in F.*

Proof Sketch. We fix variables $i, j \in [n]$ such that $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F)$ and a clause $C = C_{ij}^k \notin F$ for some $k \in [4]$. We consider only the path from (F, A) to (XOR-PAIRS$_n, \emptyset$) which has to exist, because otherwise (F, A) is redundant. There have to be claims $I_1 = (F_1, A_1)$ and $I_2 = (F_2, A_2)$ directly adjacent in this path with $F \subseteq F_1 \subseteq F_2 \subseteq \varphi$, $C \notin F_1$, $C \in F_2$, i.e. I_1 is the last claim in the path that does not contain C. I_2 is directly derived from I_1 with one of the four MICE′ steps. We can argue that this is only possible if A satisfies C. ◀

The following lemma is similar in spirit. It shows that if all clauses C_{ij} are missing in a claim, then x_i and x_j have to be set in the assumption.

▶ **Lemma 26.** *Let a* MICE′ *proof of* XOR-PAIRS$_n$ *be given and let (F, A) be an arbitrary claim in the proof. If there are $i, j \in [n]$ such that $\{x_i, x_j\} \subseteq \mathsf{vars}(F)$ and $z_{ij} \notin \mathsf{vars}(F)$, then $\{x_i, x_j\} \subseteq \mathsf{vars}(A)$.*

Proof Sketch. The proof is very similar to the one of Lemma 25. We consider a path from (F, A) to the final claim and have a closer look at the first claim in this path that contains a clause C_{ij}^k for some $k \in [4]$. We argue that we can only derive this claim if $\{x_i, x_j\} \subseteq \mathsf{vars}(A)$ is fulfilled. ◀

Using the previous two lemmas, we show that the two inference rules that multiply counts, i.e. (Join') and (Ext'), do not affect the count at all for the XOR-PAIRS formulas.

▶ **Lemma 27.** *Let a* MICE′ *proof of* XOR-PAIRS$_n$ *be given. If the proof contains a (Join') of two claims (F_1, A_1, c_1) and (F_2, A_2, c_2), then $\min(c_1, c_2) = 1$.*

Proof. Suppose otherwise, $c_1 \geq 2$ and $c_2 \geq 2$.

Assume that all x variables occurring in $\mathsf{vars}(F_1)$ are assigned in A_1. Since $c_1 \geq 2$, $\mathsf{vars}(F_1) \setminus \mathsf{vars}(A_1) \neq \emptyset$. In particular, there has to be a $z_{ij} \in \mathsf{vars}(F_1) \setminus \mathsf{vars}(A_1)$ such that there is at least one model of F_1 and A_1 with $z_{ij} = 0$ and one with $z_{ij} = 1$. Then we have $\{x_i, x_j\} \subseteq \mathsf{vars}(F_1)$ and $\{x_i, x_j\} \subseteq \mathsf{vars}(A_1)$. As a result, A_1 has to satisfy all clauses C_{ij}^k that are in F_1. Because of Lemma 25, A_1 has to satisfy the clauses C_{ij}^k that are not in F_1 as well. Thus, A_1 has to satisfy all four clauses C_{ij}^k, which is only possible if $z_{ij} \in \mathsf{vars}(A_1)$. This contradicts the choice of z_{ij}. Similarly, we also see that there is at least one x variable in $\mathsf{vars}(F_2) \setminus \mathsf{vars}(A_2)$.

Hence, we can fix $x_i \in \mathsf{vars}(F_1) \setminus \mathsf{vars}(A_1)$ and $x_j \in \mathsf{vars}(F_2) \setminus \mathsf{vars}(A_2)$. Condition (J-2) implies $x_i \notin \mathsf{vars}(F_2)$, $x_j \notin \mathsf{vars}(F_1)$ and in particular $i \neq j$. Because of $\mathsf{vars}(A_1) \subseteq \mathsf{vars}(F_1)$ and $x_j \notin \mathsf{vars}(F_1)$ we get $x_j \notin \mathsf{vars}(A_1)$ and therefore also $x_j \notin \mathsf{vars}(A_1 \cup A_2)$. The joined claim is $(F, A) = (F_1 \cup F_2, A_1 \cup A_2)$ with $\{x_i, x_j\} \subseteq \mathsf{vars}(F)$ and $C_{ij}^k \notin F$ for all k, implying $z_{ij} \notin \mathsf{vars}(F)$. With Lemma 26 we get the contradiction $x_j \in \mathsf{vars}(A) = \mathsf{vars}(A_1 \cup A_2)$.

Therefore, our assumption $c_1 \geq 2$ and $c_2 \geq 2$ was false. ◀

Using this lemma we can show, that w.l.o.g. any MICE′ proof of XOR-PAIRS$_n$ does not use (Join').

▶ **Lemma 28.** *Let π be a* MICE′ *proof of* XOR-PAIRS$_n$*. Then there is a* MICE′ *proof π' that does not use (Join′) with $c(\pi') \leq 2 \cdot c(\pi)$.*

Proof. Using π we construct a MICE′ proof π' that does not use (Join′).

For this suppose that in π, the claim $I = (F_1 \cup F_2, A_1 \cup A_2)$ is derived with (Join′) of (F_1, A_1, c_1) and (F_2, A_2, c_2). Because of Lemma 27 we can assume that $c_2 = 1$. Thus, there is a unique assignment α such that $\mathsf{vars}(A_2) \cap \mathsf{vars}(\alpha) = \emptyset$, $\mathsf{vars}(A_2 \cup \alpha) = \mathsf{vars}(F_2)$ and $A_2 \cup \alpha$ satisfies F_2. Then, we can apply (Ext′) to (F_1, A_1) resulting in $(F_1 \cup F_2, A_1 \cup A_2 \cup \alpha)$. We check the conditions to apply (Ext′).

(E-1) $F_1 \subseteq F_1 \cup F_2$ holds.

(E-2) We see that $(A_1 \cup A_2 \cup \alpha)|_{\mathsf{vars}(F_1)} = A_1|_{\mathsf{vars}(F_1)} \cup A_2|_{\mathsf{vars}(F_1)} \cup \alpha|_{\mathsf{vars}(F_1)} = A_1$. In the last equation we used three facts:

$A_1|_{\mathsf{vars}(F_1)} = A_1$ is a direct consequence of $\mathsf{vars}(A_1) \subseteq \mathsf{vars}(F_1)$.

$A_2|_{\mathsf{vars}(F_1)} \subseteq A_1$ follows from $\mathsf{vars}(A_2|_{\mathsf{vars}(F_1)}) \subseteq \mathsf{vars}(F_2) \cap \mathsf{vars}(F_1) \subseteq \mathsf{vars}(A_1)$ by (J-2) and the fact that A_1 and A_2 are consistent by (J-1).

$\alpha|_{\mathsf{vars}(F_1)} = \emptyset$. Assume otherwise that $x \in \mathsf{vars}(\alpha) \cap \mathsf{vars}(F_1)$. Then $x \in \mathsf{vars}(\alpha) \cap \mathsf{vars}(F_1) \subseteq \mathsf{vars}(F_2) \cap \mathsf{vars}(F_1) \subseteq \mathsf{vars}(A_2)$ by (J-2). Thus, $x \in \mathsf{vars}(A_2) \cap \mathsf{vars}(\alpha)$ contradicting the construction of α.

(E-3) $A_1 \cup A_2 \cup \alpha$ satisfies $(F_1 \cup F_2) \setminus F_1 \subseteq F_2$ as $A_2 \cup \alpha$ satisfies F_2 by construction.

Applying (Comp′) on the claim $(F_1 \cup F_2, A_1 \cup A_2 \cup \alpha)$ we get $(F_1 \cup F_2, A_1 \cup A_2)$. In this way we can remove every (Join′) application with one application of each (Ext′) and (Comp′). Let π' be the resulting MICE′ proof of XOR-PAIRS$_n$ that does not use (Join′). The number of claims in the proof increases at most by a factor of two. ◀

▶ **Lemma 29.** *Let a* MICE′ *proof of* XOR-PAIRS$_n$ *be given. Any claim (F, A, c) in the proof that is derived with (Ext′) from (F_1, A_1, c_1) satisfies $c = c_1$.*

Proof. Suppose $c \neq c_1$. Since $c = c_1 \cdot 2^{|\mathsf{vars}(F) \setminus (\mathsf{vars}(F_1) \cup \mathsf{vars}(A))|}$ there is a variable $v \in \mathsf{vars}(F)$ with $v \notin \mathsf{vars}(F_1) \cup \mathsf{vars}(A)$. Variable v occurs in some clause $C_{ij}^k \in F \setminus F_1$. Therefore, $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F)$. A has to satisfy all clauses of C_{ij} that occur in $F \setminus F_1$ because of (E-3). Furthermore, A has to satisfy all clauses of C_{ij} that do not occur in F as well due to Lemma 25. Since, $v \notin \mathsf{vars}(F_1)$, there is no $C_{ij} \in F_1$. Therefore, A has to satisfy all four clauses C_{ij}. For this, x_i, x_j and z_{ij} have to be set in A. Since v occurs in C_{ij}, we have $v \in \mathsf{vars}(A)$ which contradicts the choice of v. ◀

Now we have all ingredients to finally prove that the XOR-PAIRS formulas require proofs with an exponential number of MICE′ steps.

Proof of Theorem 23. Note that with Observation 24, Lemma 27 and Lemma 29 we can infer immediately that any tree-like MICE′ proof of XOR-PAIRS$_n$, i.e. any proof that uses every claim except axiom at most one time, has at least size $2^n + 2$. However, dag-like MICE′ might be stronger than tree-like MICE′. Therefore, the lower bound is not shown yet.

To prove the lower bound in the general case, let π be an arbitrary MICE′ proof of XOR-PAIRS$_n$. Let π' be a MICE′ proof of XOR-PAIRS$_n$ that does not use (Join′) with $c(\pi') \leq 2 \cdot c(\pi)$ which has to exist because of Lemma 28.

We consider an arbitrary fixed path κ in π' from the axiom to the final claim. Since π' does not use (Join′), we can only enlarge the formula with (Ext′). Because of Lemma 29, we have to assign all newly introduced variables when we use (Ext′), i.e. every variable is in at least one assumption in κ. The only rule that can remove variables from the assumption is (Comp′).

Since the final claim has the empty assumption, we have to remove all variables from the assumption in κ. Therefore, in κ there has to be at least one application of (Comp') where we remove a variable x_i from the assumption for some $i \in [n]$. Let $I_1^\kappa = (F_1^\kappa, A_1^\kappa)$ be the claim that was used for the first such (Comp') in κ to derive $I_2^\kappa = (F_2^\kappa, A_2^\kappa)$.

Let X be the set of all x variables: $X := \{x_1, \dots, x_n\}$. We show

$$X \subseteq \mathsf{vars}(F_1^\kappa).$$

Let x_i be a variable that is removed from the assumption by applying (Comp') to I_1^κ, i.e. $x_i \notin \mathsf{vars}(A_2^\kappa)$. Suppose, there is a $j \in [n]$ such that $x_j \notin \mathsf{vars}(F_1^\kappa)$ and in particular $C_{ij}^s \notin F_1^\kappa$ for all $s \in [4]$, implying $z_{ij} \notin \mathsf{vars}(F_1^\kappa)$. Let $I_r^\kappa = (F_r^\kappa, A_r^\kappa)$ be the first claim in κ with $z_{ij} \in \mathsf{vars}(F_r^\kappa)$ and therefore $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F_r^\kappa)$. I_r^κ has to be derived with (Ext'). Because of condition (E-3), A_r^κ has to satisfy all clauses C_{ij}^s in F_r^κ. Furthermore, A_r^κ has to satisfy all clauses C_{ij}^s that are not in F_r^κ because of Lemma 25. Hence, A_r^κ has to satisfy C_{ij}^s for all $s \in [4]$. To do so, we have to assign all three variables x_i, x_j and z_{ij} in A_r^κ. In particular, we have $x_i \in \mathsf{vars}(A_r^\kappa)$. Since $x_i \notin \mathsf{vars}(A_2^\kappa)$, Lemma 15 states $x_i \notin \mathsf{vars}(A_r^\kappa)$. With this contradiction we see that such an x_j with $x_j \notin \mathsf{vars}(F_1^\kappa)$ cannot exist.

Since $X \subseteq \mathsf{vars}(F_1^\kappa)$, all variables in X were introduced and assigned in the assumption with (Ext') in I_1^κ or previously in κ. Per construction there are no other (Comp') applications before I_1^κ in κ that remove variables in X. Therefore, we have

$$X \subseteq \mathsf{vars}(A_1^\kappa).$$

We show that for every $\alpha \in \mathsf{Mod}(\mathsf{XOR\text{-}PAIRS}_n)$ there is a path κ in π' with $\alpha|_X = A_1^\kappa|_X$. Assume that for some fixed model α there is no such path. Since π' does not use (Join') and $\alpha \in \mathsf{Mod}_\emptyset(\mathsf{XOR\text{-}PAIRS}_n)$, Corollary 11 implies that there is a path κ from axiom to the final claim, such that every claim (F, A) in κ fulfills $\alpha|_{\mathsf{vars}(F)} \in \mathsf{Mod}_A(F)$. In particular,

$$\alpha|_{\mathsf{vars}(F_1^\kappa)} \in \mathsf{Mod}_{A_1^\kappa}(F_1^\kappa).$$

If we restrict both sides on the variables in X and use $X \subseteq \mathsf{vars}(F_1^\kappa)$, we get

$$\alpha|_X \in \{\beta|_X \mid \beta \in \mathsf{Mod}_{A_1^\kappa}(F_1^\kappa)\}.$$

Since $X \subseteq \mathsf{vars}(A_1^\kappa)$, all models $\beta \in \mathsf{Mod}_{A_1^\kappa}(F_1^\kappa)$ have $\beta|_X = (A_1^\kappa)|_X$. Therefore, the right set has only one element which is $(A_1^\kappa)|_X$, leading to $\alpha|_X = (A_1^\kappa)|_X$. Hence, κ is a path with the claimed property for α.

Since $\mathsf{XOR\text{-}PAIRS}_n$ has 2^n models, there are (at least) 2^n paths in π' and in particular 2^n claims I_1^κ. Because every model of $\mathsf{XOR\text{-}PAIRS}_n$ assigns the x variables differently, all these claims I_1^κ are pairwise different. Therefore, π' has at least 2^n claims.

Finally, we see that the arbitrarily chosen MICE′ proof π has size $c(\pi) \geq \frac{1}{2} \cdot c(\pi') \geq 2^{n-1}$ leading to the lower bound. ◀

6 Conclusion

We performed a proof-complexity study of the #SAT proof system MICE, exhibiting hard formulas, both in terms of unsatisfiable CNFs, where their complexity in MICE matches their resolution complexity, and for highly satisfiable CNFs with many models. As Fichte et al. [18] show that MICE proofs can be extracted from solver runs for sharpSAT [29], DPDB [19] and D4 [24], this implies a number of hard instances for these #SAT solvers.

We believe that the ideas for the lower bound for our formula XOR-PAIRS can be extended to show hardness of further CNFs with many models. A natural problem for future research is to construct stronger #SAT proof systems (and #SAT solvers) where formulas such as XOR-PAIRS become easy.

As pointed out by one reviewer and mentioned in Section 1.2, there appears to be a close connection between MICE′ proofs and decision DNNFs. Therefore, it seems promising to investigate if known results from decision DNNFs can be transferred to MICE′. This may lead to more hard formulas and lower bounds for MICE′.

It would also be interesting to determine the exact relations between the systems MICE, MICE′ and the kcps(#SAT) proof system from [11] based on certified decision DNNFs.

References

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

2 Fahiem Bacchus, Shannon Dalmao, and Toniann Pitassi. Algorithms and complexity results for #sat and bayesian inference. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 340–351. IEEE Computer Society, 2003.

3 Teodora Baluta, Zheng Leong Chua, Kuldeep S. Meel, and Prateek Saxena. Scalable quantitative verification for deep neural networks. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE 2021, Madrid, Spain, 22-30 May 2021*, pages 312–323. IEEE, 2021.

4 Olaf Beyersdorff and Benjamin Böhm. Understanding the Relative Strength of QBF CDCL Solvers and QBF Resolution. In *12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, volume 185 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:20, 2021.

5 Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications. IOS Press, 2021.

6 Benjamin Böhm and Olaf Beyersdorff. Lower bounds for QCDCL via formula gauge. In Chu-Min Li and Felip Manyà, editors, *Theory and Applications of Satisfiability Testing (SAT)*, pages 47–63, Cham, 2021. Springer International Publishing.

7 Benjamin Böhm, Tomás Peitl, and Olaf Beyersdorff. QCDCL with cube learning or pure literal elimination – What is best? In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1781–1787. ijcai.org, 2022.

8 Simone Bova, Florent Capelli, Stefan Mengel, and Friedrich Slivovsky. Knowledge compilation meets communication complexity. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1008–1014. IJCAI/AAAI Press, 2016.

9 Sam Buss and Jakob Nordström. Proof complexity and SAT solving. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications, pages 233–350. IOS Press, 2021.

10 Sam Buss and Neil Thapen. DRAT proofs, propagation redundancy, and extended resolution. In Mikolás Janota and Inês Lynce, editors, *Theory and Applications of Satisfiability Testing (SAT)*, volume 11628 of *Lecture Notes in Computer Science*, pages 71–89. Springer, 2019.

11 Florent Capelli. Knowledge compilation languages as proof systems. In Mikolás Janota and Inês Lynce, editors, *Theory and Applications of Satisfiability Testing (SAT)*, volume 11628 of *Lecture Notes in Computer Science*, pages 90–99. Springer, 2019.

12 Florent Capelli, Jean-Marie Lagniez, and Pierre Marquis. Certifying top-down decision-dnnf compilers. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI) 2021*, pages 6244–6253. AAAI Press, 2021.

13 Leroy Chew and Marijn J. H. Heule. Relating existing powerful proof systems for QBF. In Kuldeep S. Meel and Ofer Strichman, editors, *25th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, volume 236 of *LIPIcs*, pages 10:1–10:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

14 Stephen A. Cook. The complexity of theorem proving procedures. In *Proc. 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

15 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, 44(1):36–50, 1979.

16 Leonardo Dueñas-Osorio, Kuldeep S. Meel, Roger Paredes, and Moshe Y. Vardi. Counting-based reliability estimation for power-transmission grids. In Satinder Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4488–4494. AAAI Press, 2017.

17 Johannes Klaus Fichte, Markus Hecher, and Florim Hamiti. The model counting competition 2020. *ACM J. Exp. Algorithmics*, 26:13:1–13:26, 2021.

18 Johannes Klaus Fichte, Markus Hecher, and Valentin Roland. Proofs for propositional model counting. In Kuldeep S. Meel and Ofer Strichman, editors, *25th International Conference on Theory and Applications of Satisfiability Testing, SAT 2022, August 2-5, 2022, Haifa, Israel*, volume 236 of *LIPIcs*, pages 30:1–30:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

19 Johannes Klaus Fichte, Markus Hecher, Patrick Thier, and Stefan Woltran. Exploiting database management systems and treewidth for counting. In Ekaterina Komendantskaya and Yanhong Annie Liu, editors, *Practical Aspects of Declarative Languages – 22nd International Symposium, PADL 2020, New Orleans, LA, USA, January 20-21, 2020, Proceedings*, volume 12007 of *Lecture Notes in Computer Science*, pages 151–167. Springer, 2020.

20 Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Model counting. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability – Second Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, pages 993–1014. IOS Press, 2021.

21 Armin Haken. The intractability of resolution. *Theor. Comput. Sci.*, 39:297–308, 1985.

22 Marijn Heule, Warren A. Hunt Jr., and Nathan Wetzler. Verifying refutations with extended resolution. In *Automated Deduction – CADE-24 – 24th International Conference on Automated Deduction*, pages 345–359, 2013.

23 Marijn J. H. Heule, Martina Seidl, and Armin Biere. Solution validation and extraction for QBF preprocessing. *J. Autom. Reason.*, 58(1):97–125, 2017.

24 Jean-Marie Lagniez and Pierre Marquis. An improved decision-dnnf compiler. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 667–673. ijcai.org, 2017.

25 Anna L. D. Latour, Behrouz Babaki, Anton Dries, Angelika Kimmig, Guy Van den Broeck, and Siegfried Nijssen. Combining stochastic constraint optimization and probabilistic programming – From knowledge compilation to constraint solving. In J. Christopher Beck, editor, *Principles and Practice of Constraint Programming – 23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28 – September 1, 2017, Proceedings*, volume 10416 of *Lecture Notes in Computer Science*, pages 495–511. Springer, 2017.

26 João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications. IOS Press, 2021.

27 Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artif. Intell.*, 175(2):512–525, 2011. doi:10.1016/j.artint.2010.10.002.

28 Weijia Shi, Andy Shih, Adnan Darwiche, and Arthur Choi. On tractable representations of binary neural networks. In Diego Calvanese, Esra Erdem, and Michael Thielscher, editors, *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020*, pages 882–892, 2020.

29 Marc Thurley. sharpSAT – Counting models with advanced component caching and implicit BCP. In Armin Biere and Carla P. Gomes, editors, *Theory and Applications of Satisfiability Testing – SAT 2006, 9th International Conference, Seattle, WA, USA, August 12-15, 2006, Proceedings*, volume 4121 of *Lecture Notes in Computer Science*, pages 424–429. Springer, 2006.

30 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.

31 Moshe Y. Vardi. Boolean satisfiability: Theory and engineering. *Commun. ACM*, 57(3):5, 2014.

32 Marc Vinyals. Hard examples for common variable decision heuristics. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

33 Nathan Wetzler, Marijn Heule, and Warren A. Hunt Jr. Drat-trim: Efficient checking and trimming using expressive clausal proofs. In Carsten Sinz and Uwe Egly, editors, *Theory and Applications of Satisfiability Testing (SAT*, volume 8561 of *Lecture Notes in Computer Science*, pages 422–429. Springer, 2014.

34 Ennan Zhai, Ang Chen, Ruzica Piskac, Mahesh Balakrishnan, Bingchuan Tian, Bo Song, and Haoliang Zhang. Check before you change: Preventing correlated failures in service updates. In Ranjita Bhagwan and George Porter, editors, *17th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2020, Santa Clara, CA, USA, February 25-27, 2020*, pages 575–589. USENIX Association, 2020.