Journal on Satisfiability, Boolean Modeling and Computation 15 (2024) 27–59

Proof Complexity of Propositional Model Counting

Olaf Beyersdorff olaf.beyersdorff@uni-jena.de
Tim Hoffmann hoffmann.t@uni-jena.de
Luc N. Spachmann luc.spachmann@uni-jena.de
Friedrich Schiller University 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. We then establish an exponential lower bound for the number of proof steps in MICE′ (and hence also in MICE) for a specific family of CNFs. We also explain a tight connection between MICE′ proofs and decision DNNFs.

Keywords: *Model counting, #SAT, proof complexity, proof systems, lower bounds*

Submitted 22 November 2023; revised 15 October 2024; accepted 22 October 2024

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 [19], making it intractable unless P = NP. From the practical side, the 'SAT revolution' [37] with the evolution of practical SAT solvers has turned SAT into a tractable problem for many industrial instances [8].

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 [36] 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 [26] 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 [23]. #SAT solvers allow to tackle a large variety of real-world questions, including all kinds of problems in the areas probabilistic reasoning [2,31], risk analysis [22,40] and explainable artificial intelligence [3,34].

Unlike in SAT solving where conflict-driven clause learning (CDCL) [32] dominates the scene, there are a number of conceptually different approaches to #SAT solving, including the lifting of standard techniques from SAT-solving [35], employing knowledge compilation [30], and via dynamic programming [25]. 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 [14]. While we know that in principle every SAT solver implicitly defines a proof system, a seminal result of [1,33] 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 [38]. 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 [6,9,10]. 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 [28,39] and QRAT for QBF [29] for this purpose. These proof systems aim to capture all modern solving techniques, including preprocessing and therefore tend to be very powerful [15,18]. 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 three 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) [16]. A very similar idea was used to modify current knowledge compilers such that they output *certifiable* decision DNNFs [17]. With the help of an implemented checker it can be verified in polynomial time that a given CNF is indeed equivalent to the resulting certifiable decision DNNF.

The second, a line based proof system called MICE [24] (the acronym stands for Model-counting Induction by Claim Extension), was introduced in 2022 [24]. 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 [35], DPDB [25] and D4 [30], as demonstrated in [24]. Hence MICE proofs can be used to verify the correctness of answers of these #SAT solvers.

A third proof system was introduced very recently [13] for certified proof checking of #SAT solvers. The system is similar in spirit to the general proof-checking formats RAT and DRAT [28,39] used for SAT solving and employs Partitioned-Operation Graphs (POGs).

1.1. Our Contributions

We perform a proof complexity analysis of the #SAT proof system MICE from [24]. 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 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 4.8 and 4.9). Hence also MICE' can be used for proof logging for the #SAT solvers mentioned above.

(b) Lower bounds for MICE and MICE'. 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 5.1).

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 $\mathsf{XOR\text{-}PAIRS}_n$, based on the parity function (Theorem 5.6). Technically, our lower bound is established by showing that in MICE' proofs of $\mathsf{XOR\text{-}PAIRS}_n$, all applications of the join and extension rules preserve the model count.

(c) A connection between MICE and decision DNNFs. We show a tight connection between MICE' and decision DNNFs. Specifically, we efficiently extract a decision DNNF from a MICE' proof (Theorem 6.1). This provides an alternative way to obtain lower bounds for MICE'.

1.2. Organisation

The remainder of this article 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 [24] 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 results on the exponential lower bound for MICE' (and hence for MICE). In Section 6 we explain the connections to decision DNNFs, yielding additional MICE' lower bounds. We conclude in Section 7 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 [20]. 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 $\Box$. 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 [24] and show some basic properties of the system.

Definition 3.1 ([24])**.** 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 ($\mathsf{1\text{-}Mod}$), Composition (Comp), Join (Join) and Extension (Ext). They are specified in Fig. 1. We give some intuition on the rules. The axiom rule ($\mathsf{1\text{-}Mod}$) 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.* We want to emphasize that the rule (Comp) can be applied with $n = 0$, i.e. we can derive

Exactly One Model.

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

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

Composition.

$$\frac{((F,V),A_1,c_1) \quad \cdots \quad ((F,V),A_n,c_n)}{((F,V),A,\sum_{i\in[n]} c_i)} \quad (\mathsf{Comp})$$

- (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 $A \cup \{C|_V \mid C \in F\} \cup \{\overline{A}_i \mid i \in [n]\}$. Such a refutation is included into the trace and is called an *absence of models statement.*

Join.

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

- (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.

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

- (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 [24].

any claim $((F,V),A,0)$ if $A \cup \{C|_V \mid C \in F\}$ is unsatisfiable. In particular, we can derive $((\varphi,\mathsf{vars}(\varphi)),\emptyset,0)$ for any unsatisfiable formula φ with a single application of (Comp).

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 3.2 (Fichte, Hecher, Roland [24])**.** 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 ($\mathsf{1\text{-}Mod}$), (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 [24] 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.

In a correct claim $((F, V), A, c)$ the count c is uniquely determined by 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 sequence of claims $I_1, \dots, I_m$ 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 3.3. *If some assumption A satisfies an arbitrary formula F with $\mathsf{vars}(A) \subseteq \mathsf{vars}(F)$, 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 $\mathsf{vars}(F) \setminus \mathsf{vars}(A) = \{x_1, \dots, x_n\}$. 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 ($\mathsf{1\text{-}Mod}$). 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 $\rho_i = ((x_i), (\overline{x}_i), \Box)$. We use ($\mathsf{Join}$) 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 ($\mathsf{Ext}$) 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 3.4. *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 3.1 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 3.6 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 3.5. *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: 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 argue that this is not possible:

- It is impossible with (1-Mod), since this rule uses no previous claim.
- It is impossible with (Comp), since $V_1 \neq V_2$.
- It is impossible with (Join). Assume otherwise that $((F_1, V_1), A_1)$ is joined with some $((F_3, V_3), A_3)$. Because of $x \in V_2 = V_1 \cup V_3$ we have $x \in V_3$. Then $x \in \mathsf{vars}(F_1) \cap (V_3 \setminus V_1)$, contradicting condition (J-3).
- It is impossible with (Ext). Otherwise x has to be in $\mathsf{vars}(A_2)$ because of (E-2) and $x \in V_2 \setminus V_1$ per construction. Then $A_2|_{V_2 \setminus V_1}$ satisfies a clause in F_1 since both literals x and $\overline{x}$ occur in F_1 (because $F \subseteq F_1$). Thus condition (E-5) fails.

This leads to a contradiction. As a result, such an x can not exist. ☐

Lemma 3.6. *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: Let $\pi = (I_1, \ldots, I_m)$ with $I_i = ((F_i, V_i), A_i)$. Because of Lemma 3.5, 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

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

with the unique α_i defined above. Note that A_i and α_i have no variables in common and are therefore consistent. The resulting claim on the right side satisfies the requirement we want to achieve.

We show by induction that $(f(I_1), \ldots, f(I_k))$ is a valid MICE trace for all $k \in \{0, \ldots, m\}$. In the base case $k = 0$ the empty trace is valid. For the induction step we assume that we have already derived $f(I_1), \ldots, f(I_{k-1})$. In particular, we have derived $f(I)$ for every claim I we used to derive I_k. We consider the different rules from which I_k could be derived.
Exactly One Model. $I_k = ((F_k, V_k), A_k)$ is derived with (1-Mod). We can derive $f(I_k) = ((F_k, V_k \cup \mathsf{vars}(F_k)), A_k \cup \alpha_k)$ with (1-Mod) as well.

- (O-1). $\mathsf{vars}(A_k \cup \alpha_k) = V_k \cup \mathsf{vars}(F_k)$ since $\mathsf{vars}(A_k) = V_k$ ((O-1) for I_k) and $\mathsf{vars}(\alpha_k) = \mathsf{vars}(F_k) \setminus V_k$.
- (O-2). $A_k \cup \alpha_k$ satisfies F_k, since A_k satisfies F_k ((O-2) for I_k).

Composition. $I_k = ((F_k, V_k), A_k)$ is derived with (Comp) of claims $I_{i_1}, \ldots, I_{i_r}$ with $i_j < k$ and $I_{i_j} = ((F_k, V_k), A_{i_j})$ for $j \in [r]$. Let ρ be the absence of models statement ((C-3) for I_k) that refutes

$$\{C|_{V_k} \mid C \in F_k\} \cup A_k \cup \{\overline{A}_{i_j} \mid j \in [r]\}.$$

For $j \in [r]$ let $f(I_{i_j}) = ((F_k, V_k \cup \mathsf{vars}(F_k)), A_{i_j} \cup \alpha_k)$ with $\alpha_k = \alpha_{i_j}$, since α_{i_j} does only depend on F_k and V_k and is therefore equal to α_k. To derive $f(I_k) = ((F_k, V_k \cup \mathsf{vars}(F_k)), A_k \cup \alpha_k)$ we can use (Comp) of $f(I_{i_1}), \ldots, f(I_{i_r})$:

- (C-1). $A_{i_j} \cup \alpha_k$ assign the same variables and are pairwise inconsistent, since A_{i_j} assign the same variables and are pairwise inconsistent ((C-1) for I_k).
- (C-2). For every $j \in [r]$ we have $A_k \subseteq A_{i_j}$ ((C-2) for I_k) and in particular $A_k \cup \alpha_k \subseteq A_{i_j} \cup \alpha_k$.
- (C-3). We need an absence of models statement that refutes

  $$\begin{aligned} &\{C|_{V_k \cup \mathsf{vars}(F_k)} \mid C \in F_k\} \cup (A_k \cup \alpha_k) \cup \{\overline{A_{i_j} \cup \alpha_k} \mid j \in [r]\} \\ &\quad = F_k \cup A_k \cup \alpha_k \cup \{(\overline{A}_{i_j} \vee \overline{\alpha}_k) \mid j \in [r]\} \end{aligned}$$

  For this we do at most $(|F_k| + r) \cdot |\alpha_k| = O(|\pi|^2)$ resolution steps to remove all α_k literals from F_k and $(\overline{A}_{i_j} \vee \overline{\alpha}_k)$. Note that this is possible, since for any $x \in \mathsf{lits}(\alpha_k)$, only $\overline{x}$ can appear in $\mathsf{lits}(F_k)$ per construction of α_k. It remains exactly the formula that is refuted by ρ as all variables from α_k are removed.

Join. $I_k = ((F_k, V_k), A_k) = ((F_i \cup F_j, V_i \cup V_j), A_i \cup A_j)$ is derived using (Join) of claims $I_i = ((F_i, V_i), A_i)$ and $I_j = ((F_j, V_j), A_j)$ with $i, j < k$. First, we show

$$\mathsf{vars}(F_i) \setminus (V_i \cup V_j) = \mathsf{vars}(F_i) \setminus V_i.$$

The inclusion $\subseteq$ follows directly. To show the other direction $\supseteq$, assume $x \in \mathsf{vars}(F_i)$ and $x \notin V_i$. Because of (J-3) for I_k is $x \notin \mathsf{vars}(F_i) \cap (V_j \setminus V_i)$. Thus, $x \notin V_j$ and therefore, $x \in \mathsf{vars}(F_i) \setminus (V_i \cup V_j)$.

Using this we can prove

$$\alpha_k = \alpha_i \cup \alpha_j.$$

For that it is sufficient to show that both sides assign the same variables and that they are consistent.

We show that $\mathsf{vars}(\alpha_k) = \mathsf{vars}(\alpha_i) \cup \mathsf{vars}(\alpha_j)$. With the definitions of α and F_k we get $\mathsf{vars}(\alpha_k) = \mathsf{vars}(F_k) \setminus V_k = \mathsf{vars}(F_i \cup F_j) \setminus (V_i \cup V_j)$. Applying simple set operations and the equation from above, this is equal to $(\mathsf{vars}(F_i) \setminus (V_i \cup V_j)) \cup (\mathsf{vars}(F_j) \setminus (V_i \cup V_j)) = (\mathsf{vars}(F_i) \setminus V_i) \cup (\mathsf{vars}(F_j) \setminus V_j)$. This is exactly $\mathsf{vars}(\alpha_i) \cup \mathsf{vars}(\alpha_j)$ per definition.

To show consistency of α_i, α_j and α_k, we show that every pair is consistent. α_i and α_j are consistent: Otherwise suppose $x \in \mathsf{lits}(\alpha_i)$ and $\overline{x} \in \mathsf{lits}(\alpha_j)$ for some literal x. Per construction is $\overline{x} \in \mathsf{lits}(F_i)$, $x \in \mathsf{lits}(F_j)$ and therefore, $x \in \mathsf{lits}(F_k)$, $\overline{x} \in \mathsf{lits}(F_k)$. Furthermore, $\mathsf{var}(x) \notin (V_i \cup V_j) = V_k$. As a result, $\mathsf{var}(x) \in \mathsf{vars}(F_k) \setminus V_k$ and x occurs in both polarities in F_k leading to a contradiction to Lemma 3.5. α_k and α_i are consistent: Assume $x \in \mathsf{vars}(\alpha_k)$ and $x \in \mathsf{vars}(\alpha_i)$ for some variable x. W.l.o.g. let $x \in \mathsf{lits}(\alpha_k)$. Then, $\overline{x} \in \mathsf{lits}(F_k) = \mathsf{lits}(F_i \cup F_j)$ leading to $\overline{x} \in \mathsf{lits}(F_i)$ and $x \in \mathsf{lits}(\alpha_i)$. Analogously we get that α_k and α_j are consistent.

To derive

$$\begin{aligned} f(I_k) &= ((F_k, V_k \cup \mathsf{vars}(F_k)), A_k \cup \alpha_k) \\ &= ((F_i \cup F_j, V_i \cup V_j \cup \mathsf{vars}(F_i) \cup \mathsf{vars}(F_j)), A_i \cup A_j \cup \alpha_i \cup \alpha_j) \end{aligned}$$

we can use (Join) of $f(I_i) = ((F_i, V_i \cup \mathsf{vars}(F_i)), A_i \cup \alpha_i)$ and $f(I_j) = ((F_j, V_j \cup \mathsf{vars}(F_j)), A_j \cup \alpha_j)$:

- (J-1). A_i and A_j are consistent because of (J-1) for I_k. We showed already that α_i and α_j are consistent. A_i and α_j are consistent, because they have no variables in common. Otherwise let x be a variable with $x \in \mathsf{vars}(A_i)$ and $x \in \mathsf{vars}(\alpha_j)$. Per construction is $x \in V_i$, $x \in \mathsf{vars}(F_j)$ and $x \notin V_j$ and thus, $x \in \mathsf{vars}(F_j) \cap (V_i \setminus V_j)$. This contradicts (J-3) for I_k. The same argument shows that A_j and α_i are consistent.
  As a result, $A_i \cup \alpha_i$ and $A_j \cup \alpha_j$ are consistent.
- (J-2). First we show

$$V_i \cap \mathsf{vars}(\alpha_j) = \emptyset \quad \text{and} \quad V_j \cap \mathsf{vars}(\alpha_i) = \emptyset.$$

  For the sake of contradiction, assume there is a variable x with $x \in V_i$ and $x \in \mathsf{vars}(\alpha_j)$. Per construction is $x \in \mathsf{vars}(F_j)$ and $x \notin V_j$ and thus $x \in \mathsf{vars}(F_j) \cap (V_i \setminus V_j)$ which contradicts (J-3) for I_k. Analogously we get $V_j \cap \mathsf{vars}(\alpha_i) = \emptyset$. Furthermore, $(V_i \cap V_j) \subseteq \mathsf{vars}(A_r)$ for $r \in \{i, j\}$ because of (J-2) for I_k. Using this, we get

$$\begin{aligned}
&\big(V_i \cup \mathsf{vars}(\alpha_i)\big) \cap \big(V_j \cup \mathsf{vars}(\alpha_j)\big) \\
&\quad = (V_i \cap V_j) \cup \big(\mathsf{vars}(\alpha_i) \cap \mathsf{vars}(\alpha_j)\big) \cup \big(V_i \cap \mathsf{vars}(\alpha_j)\big) \cup \big(V_j \cap \mathsf{vars}(\alpha_i)\big) \\
&\quad = (V_i \cap V_j) \cup \big(\mathsf{vars}(\alpha_i) \cap \mathsf{vars}(\alpha_j)\big) \\
&\quad \subseteq \mathsf{vars}(A_r) \cup \big(\mathsf{vars}(\alpha_i) \cap \mathsf{vars}(\alpha_j)\big) \\
&\quad \subseteq \mathsf{vars}(A_r) \cup \mathsf{vars}(\alpha_r)
\end{aligned}$$

  for $r \in \{i, j\}$.
- (J-3). The requirement $\mathsf{vars}(F_i) \cap ((V_i \cup V_j) \setminus V_i) = \emptyset$ is always fulfilled if the two joined claims satisfy $\mathsf{vars}(F_i) \subseteq V_i$.

Extension. $I_k = ((F_k, V_k), A_k)$ is derived using (Ext) of $I_i = ((F_i, V_i), A_i)$ with $i < k$. Then we can also derive $f(I_k) = ((F_k, V_k \cup \mathsf{vars}(F_k)), A_k \cup \alpha_k)$ from $f(I_i) = ((F_i, V_i \cup \mathsf{vars}(F_i)), A_i \cup \alpha_i)$ with (Ext):

- (E-1). $F_i \subseteq F_k$, $V_i \cup \mathsf{vars}(F_i) \subseteq V_k \cup \mathsf{vars}(F_k)$ is fulfilled, since $F_i \subseteq F_k$ and $V_i \subseteq V_k$ because of (E-1) for I_k.
- (E-2). We have to show $(V_k \cup \mathsf{vars}(F_k)) \setminus (V_i \cup \mathsf{vars}(F_i)) \subseteq \mathsf{vars}(A_k \cup \alpha_k)$. For this, let x be an arbitrary variable with $x \in (V_k \cup \mathsf{vars}(F_k)) \setminus (V_i \cup \mathsf{vars}(F_i))$. If $x \in V_k$, then $x \in V_k \setminus V_i \subseteq \mathsf{vars}(A_k)$ because of (E-2) for I_k. Otherwise if $x \notin V_k$, $x \in \mathsf{vars}(F_k)$ and thus $x \in \mathsf{vars}(\alpha_k)$ per construction of α_k.
- (E-3). We have to show that $(A_k \cup \alpha_k)|_{V_i \cup \mathsf{vars}(F_i)} = A_i \cup \alpha_i$. For this we use

$$A_k|_{V_i} = A_i$$

  which follows from (E-3) for I_k. Furthermore, by using $V_i \subseteq V_k$ and $\mathsf{vars}(\alpha) \cap V_k = \emptyset$, we receive

$$\alpha_k|_{V_i} = \alpha_k|_{V_k \cap V_i} = (\alpha_k|_{V_k})|_{V_i} = \emptyset.$$

  Next, we prove

$$(A_k \cup \alpha_k)|_{\mathsf{vars}(F_i) \setminus V_i} = \alpha_i.$$

For this we show that both assign the same variables and then that every variable is assigned equally.
To show that both sides assign the same variables, the direction $\subseteq$ follows with $\mathsf{vars}((A_k \cup \alpha_k)|_{\mathsf{vars}(F_i)\setminus V_i}) \subseteq \mathsf{vars}(F_i) \setminus V_i = \mathsf{vars}(\alpha_i)$. For the other direction $\supseteq$, let $x \in \mathsf{vars}(\alpha_i)$ implying $x \in \mathsf{vars}(F_i) \setminus V_i$. Thus, we have to show that $x \in \mathsf{vars}(A_k \cup \alpha_k)$. If $x \in V_k$, then $x \in V_k \setminus V_i$ and thus, $x \in \mathsf{vars}(A_k)$ because of (E-2) for I_k. If $x \notin V_k$, then $x \in \alpha_k$, since $x \in \mathsf{vars}(F_i) \subseteq \mathsf{vars}(F_k)$.
In order to show that α_k and α_i are consistent, assume $x \in \mathsf{vars}(\alpha_k) \cap \mathsf{vars}(\alpha_i)$ and let $x \in \mathsf{lits}(\alpha_i)$. Then we have $\overline{x} \in \mathsf{lits}(F_i) \subseteq \mathsf{lits}(F_k)$ leading to $x \in \mathsf{lits}(\alpha_k)$. A_k, α_i are consistent: Assume $x \in \mathsf{vars}(A_k) \cap \mathsf{vars}(\alpha_i)$ and let $x \in \mathsf{lits}(\alpha_i)$. Then $\overline{x} \in \mathsf{lits}(F_i)$, $x \notin V_i$, $x \in V_k$. Because of (E-5) for I_k, $A_k|_{V_k\setminus V_i}$ and in particular $A_k|_{\{x\}}$ does not satisfy any $C \in F_i$. Since there is a clause in F_i that contains literal $\overline{x}$, $x \in \mathsf{lits}(A_k)$.
Using those three properties from above we get

$$(A_k \cup \alpha_k)|_{V_i \cup \mathsf{vars}(F_i)} = A_k|_{V_i} \cup \alpha_k|_{V_i} \cup (A_k \cup \alpha_k)|_{\mathsf{vars}(F_i)\setminus V_i} = A_i \cup \alpha_i.$$

- (E-4). $(A_k \cup \alpha_k)$ satisfies $F_k \setminus F_i$, since A_k satisfies $F_k \setminus F_i$ (E-4) for I_k.
- (E-5). $(A_k \cup \alpha_k)|_{(V_k \cup \mathsf{vars}(F_k))\setminus(V_i \cup \mathsf{vars}(F_i))}$ does not satisfy C for any $C \in F_i$ as the restricted assignment has no variables in $\mathsf{vars}(F_i)$.

This completes the induction. Since $I_m = ((\varphi, \mathsf{vars}(\varphi)), \emptyset) = f(I_m)$, $\pi' = (f(I_1), \ldots, f(I_m))$ is a valid proof for φ with the claimed property. The number of claims does not change. The number of clauses in the refutation does only increase in the (Comp) case and at most by a factor of $O(s(\pi)^2)$. ☐

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 3.7. *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-3) 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 3.8. *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: Consider the family of formulas $(T_n)_{n\in\mathbb{N}}$ that only have one clause

$$(x_1 \vee x_2 \vee \cdots \vee x_n).$$

First we show that T_n has a MICE proof of size $O(n^2)$ for every n. With the construction of Proposition 3.3 we derive

$$I_1 = ((T_n, \mathsf{vars}(T_n)), \{x_1 = 1\}, 2^{n-1}),$$
$$I_2 = ((T_n, \mathsf{vars}(T_n)), \{x_1 = 0, x_2 = 1\}, 2^{n-2}),$$
$$\vdots$$
$$I_n = ((T_n, \mathsf{vars}(T_n)), \{x_1 = 0, x_2 = 0, \ldots, x_{n-1} = 0, x_n = 1\}, 1).$$

We apply (Comp) to the one claim I_n which results in

$$J_n = ((T_n, \mathsf{vars}(T_n)), \{x_1 = 0, x_2 = 0, \ldots, x_{n-1} = 0\}, 1).$$

Then, we use (Comp) of J_n and I_{n-1} which results in

$$J_{n-1} = ((T_n, \mathsf{vars}(T_n)), \{x_1 = 0, x_2 = 0, \ldots, x_{n-2} = 0\}, 3).$$

Similarly we apply (Comp) to every pair of claims I_i and J_{i+1} and finally get

$$J_1 = ((T_n, \mathsf{vars}(T_n)), \emptyset, 2^n - 1).$$

In total we need $O(n^2)$ steps to derive all I_i and n applications of (Comp) to combine these claims.

Next, we show that any MICE proof with the additional requirement $\mathsf{vars}(F) = V$ has size $2^{\Omega(n)}$. Note that the construction from Proposition 3.3 does not work under this additional requirement.

We show that the claim $I_\emptyset = ((\emptyset, \emptyset), \emptyset, 1)$ does not help for the proof. If we use (Join) on $I_\emptyset$ together with any claim I, the result is I. Similarly, if we derive I with (Ext) from $I_\emptyset$, we can derive I with (1-Mod) without $I_\emptyset$. If we apply (Comp) on claim $I_\emptyset$ together with some other claims, (C-1) implies that all used claims have to be $I_\emptyset$. Thus, (Comp) would result in $I_\emptyset$. Therefore, we can assume $I_\emptyset$ is not in the proof at all.

Thus, the only component we can use is $C = (\{x_1 \vee \ldots \vee x_n\}, \{x_1, \ldots, x_n\})$. Assume I is derived with (Join) from $I_1 = (C, A_1)$ and $I_2 = (C, A_2)$. Condition (J-2) implies $\mathsf{vars}(A_1) = \mathsf{vars}(A_2) = \{x_1, \ldots, x_n\}$ and in particular $A_1 = A_2$ because of (J-1). Therefore, $I = I_1 = I_2$ and the usage of (Join) is redundant. Let $I = (C, A)$ be derived from $I_1 = (C, A_1)$ with (Ext). Because of (E-3) we have $A = A_1$ and hence $I = I_1$. Hence, the rules (Join) and (Ext) achieve nothing and we can assume that they do not appear in the proof.

As a result, the proof can only use rules (1-Mod) and (Comp). Such a proof needs $2^n - 1$ applications of (1-Mod) as T_n has $2^n - 1$ models. □

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.

Axiom.

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

Composition.

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

- (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.

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

- (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 $\mathsf{MICE'}$.

The rules of $\mathsf{MICE'}$ are Axiom (Ax), Composition ($\mathsf{Comp'}$), Join ($\mathsf{Join'}$) and Extension ($\mathsf{Ext'}$). They are specified in Fig. 2.

The intuition for the rules ($\mathsf{Comp'}$) and ($\mathsf{Join'}$) 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. ($\mathsf{Ext'}$) 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 a result, we extend one model of the original claim to one new model. In ($\mathsf{Ext'}$) 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.

Definition 4.1 (Adapted Proof System $\mathsf{MICE'}$)**.** 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 $\mathsf{MICE'}$ are (Ax), ($\mathsf{Comp'}$), ($\mathsf{Join'}$) and ($\mathsf{Ext'}$). 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 4.2. *The inference rules of* MICE' *are sound.*

Proof: To prove the soundness of every MICE' rule, we associate every claim (F, A, c) with the set $\mathsf{Mod}_A(F)$. 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 $|\mathsf{Mod}_\emptyset(\emptyset)| = |\{\emptyset\}| = 1$.

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

$$\begin{aligned}
&\mathsf{Mod}_A(F)\\
&\quad = \{\alpha \in \langle \mathsf{vars}(F)\rangle \mid \alpha \models F \cup A\}\\
&\quad = \biguplus_{i\in[n]} \{\alpha \in \langle \mathsf{vars}(F)\rangle \mid \alpha \models F \cup A_i\} \uplus \{\alpha \in \langle \mathsf{vars}(F)\rangle \mid \alpha \models F \cup A \cup \{\overline{A}_i \mid i \in [n]\}\}
\end{aligned}$$

where $\uplus$ denotes the disjoint union. This split of A into those A_i is possible since $A \subseteq A_i$ (C-2). The sets on the right side of the equation are pairwise disjoint because of (C-1). The last set is empty, otherwise there would not exist an absence of models statement (C-3). Thus,

$$\mathsf{Mod}_A(F) = \biguplus_{i\in[n]} \mathsf{Mod}_{A_i}(F).$$

Using the correctness of all used claims we get

$$|\mathsf{Mod}_A(F)| = \sum_{i\in[n]} |\mathsf{Mod}_{A_i}(F)| = \sum_{i\in[n]} c_i.$$

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 show that

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

We will prove both subset relations separately in the following.

For $\subseteq$, let $\alpha \in \mathsf{Mod}_{A_1\cup A_2}(F_1 \cup F_2)$ be given. Per definition, α satisfies $F_1 \cup F_2 \cup A_1 \cup A_2$ and in particular $\alpha|_{\mathsf{vars}(F_1)\cup\mathsf{vars}(A_1)}$ has to satisfy $F_1 \cup A_1$. Because of $\mathsf{vars}(A_1) \subseteq \mathsf{vars}(F_1)$, $\alpha|_{\mathsf{vars}(F_1)}$ has to satisfy $F_1 \cup A_1$ and therefore, $\alpha|_{\mathsf{vars}(F_1)} \in \mathsf{Mod}_{A_1}(F_1)$. Analogously, we get $\alpha|_{\mathsf{vars}(F_2)} \in \mathsf{Mod}_{A_2}(F_2)$. Since $\alpha = \alpha|_{\mathsf{vars}(F_1)} \cup \alpha|_{\mathsf{vars}(F_2)}$, we can choose $\alpha_1 = \alpha|_{\mathsf{vars}(F_1)}$ and $\alpha_2 = \alpha|_{\mathsf{vars}(F_2)}$ to see the relation.

For the other direction $\supseteq$, we first have to show that any fixed $\alpha_1 \in \mathsf{Mod}_{A_1}(F_1)$ and $\alpha_2 \in \mathsf{Mod}_{A_2}(F_2)$ are consistent. Because of (J-2) ensuring $\mathsf{vars}(F_1) \cap \mathsf{vars}(F_2) \subseteq \mathsf{vars}(A_i)$ for both $i \in \{1, 2\}$, we know that they could only be inconsistent in variables in A_i. With (J-1) which states that A_1 and A_2 are consistent, we can conclude that α_1 and α_2 are consistent. We know that α_i satisfies $F_i \cup A_i$ per construction. As a result, $\alpha_1 \cup \alpha_2$ satisfies $F_1 \cup F_2 \cup A_1 \cup A_2$ and is therefore in $\mathsf{Mod}_{A_1 \cup A_2}(F_1 \cup F_2)$.

The model count for the derived claim follows directly with the correctness of both used claims,

$$\left|\mathsf{Mod}_{A_1 \cup A_2}(F_1 \cup F_2)\right| = \left|\mathsf{Mod}_{A_1}(F_1)\right| \cdot \left|\mathsf{Mod}_{A_2}(F_2)\right| = c_1 \cdot c_2.$$

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 show

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

Similarly to the previous case, we prove both inclusions separately.

For $\subseteq$, let $\gamma \in \mathsf{Mod}_A(F)$ be given. Per definition, γ satisfies $F \cup A = F_1 \cup (F \setminus F_1) \cup A_1 \cup (A \setminus A_1)$. This split is possible because of (E-1) and (E-2). We can define $\alpha = \gamma|_{\mathsf{vars}(F_1)}$, $\beta = \gamma|_{\mathsf{vars}(F) \setminus (\mathsf{vars}(F_1) \cup \mathsf{vars}(A))}$. Then we have $\gamma = \alpha \cup (A \setminus A_1) \cup \beta$ and get the inclusion.

For $\supseteq$, we fix some $\alpha \in \mathsf{Mod}_{A_1}(F_1)$, $\beta \in \langle \mathsf{vars}(F) \setminus (\mathsf{vars}(F_1) \cup \mathsf{vars}(A)) \rangle$ and define $\gamma = \alpha \cup (A \setminus A_1)$. As α has to contain the assignment according to A_1, we have that γ satisfies A. With (E-3) follows that γ satisfies $F \setminus F_1$. Since A_1 is a model of F_1, γ satisfies F_1 as well. As a result, γ satisfies $F \cup A$ and is therefore in $\mathsf{Mod}_A(F)$.

The corresponding model count follows immediately with the correctness of (F_1, A_1, c_1),

$$\left|\mathsf{Mod}_A(F)\right| = \left|\mathsf{Mod}_{A_1}(F_1)\right| \cdot \left|\mathsf{Mod}_A(F \setminus F_1)\right| = c_1 \cdot 2^{|\mathsf{vars}(F) \setminus (\mathsf{vars}(F_1) \cup \mathsf{vars}(A))|}.$$

As we have shown with the easy semantic arguments above, all rules of MICE' are sound. $\square$

Corollary 4.3. *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 3.3.

Definition 4.4 (Satisfying Assumption Rule)**.** Under the condition (S-1): A satisfies F, we allow to derive

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

This rule is sound and does not make MICE' proofs much shorter. Therefore, when constructing MICE' proofs, we sometimes use this additional rule as it makes proofs more intuitive and easier to understand.

Lemma 4.5. *(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. $\square$

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

Theorem 4.6. 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 4.2.

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. $\square$

Next, we show some basic properties of MICE'.

Lemma 4.7. *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 $((F_1, A_1), \dots, (F_n, A_n) = (F, A))$ 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. $\square$

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 4.8. MICE' *p-simulates* MICE.

Proof: Let $\pi = (I_1, \dots, I_m)$ 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 3.6. We will show that for $f(((F, V), A)) := (F, A|_{\mathsf{vars}(F)})$ the sequence $\pi' = (f(I_1), \dots, f(I_m))$ is a correct MICE' proof of φ.

For this we first prove by induction that $(f(I_1), \dots, f(I_k))$ is a MICE' proof trace for every $k \in \{0, \dots, m\}$. In the base case $k = 0$ the empty trace is valid. For the induction step we assume we have already derived $f(I_1), \dots, f(I_{k-1})$ and in particular $f(I)$ for all claims I we used to derive I_k. We distinguish how I_k is derived.

Exactly One Model. $I_k = ((F, V), A)$ is derived with $(\mathsf{1\text{-}Mod})$. Then we can derive $f(I_k) = (F, A|_{\mathsf{vars}(F)})$ with (SA) since A satisfies F ((O-2) for I_k) and in particular, $A|_{\mathsf{vars}(F)}$ satisfies F.

Composition. $I_k = ((F, V), A)$ is derived with (Comp) using absence of models statement ρ and claims $I_{i_1}, \dots, I_{i_r}$ for $i_j < k$ with $I_{i_j} = ((F, V), A_{i_j})$ and $f(I_{i_j}) = (F, A_{i_j}|_{\mathsf{vars}(F)})$. Note that some $f(I_{i_j})$ might be duplicates. We can derive $f(I_k) = (F, A|_{\mathsf{vars}(F)})$ with (Comp') of claims $f(I_{i_j})$ after removing all duplicates:

- (C-1). $A_{i_j}|_{\mathsf{vars}(F)}$ assign the same variables, since A_{i_j} assign the same variables ((C-1) for I_k). The new assumptions are pairwise inconsistent as we removed all duplicates.
- (C-2). $A|_{\mathsf{vars}(F)} \subseteq A_{i_j}|_{\mathsf{vars}(F)}$ follows from $A \subseteq A_{i_j}$ ((C-2) for I_k).
- (C-3). There is an absence of models statement ρ ((C-3) for I_k) which is a refutation of

$$A \cup F \cup \{\overline{A}_{i_j} \mid j \in [r]\}$$

where we used our assumption $\mathsf{vars}(F) \subseteq V$. ρ can be adapted to a refutation of

$$A|_{\mathsf{vars}(F)} \cup F \cup \{\overline{A}_{i_j}|_{\mathsf{vars}(F)} \mid j \in [r]\},$$

since we can just remove the variables that are not in $\mathsf{vars}(F)$ from every clause in ρ and get a valid resolution proof where some resolutions might get weakening steps.

Join. $I_k = ((F_i \cup F_j, V_i \cup V_j), A_i \cup A_j)$ is derived with (Join) using claims I_i and I_j, with $i, j < k$. For $r \in \{i, j\}$ let $I_r = ((F_r, V_r), A_r)$ and $f(I_r) = (F_r, A_r|_{\mathsf{vars}(F_r)})$. We can apply (Join') to $f(I_i)$ and $f(I_j)$:

- (J-1). $A_i|_{\mathsf{vars}(F_i)}$ and $A_j|_{\mathsf{vars}(F_j)}$ are consistent since A_i and A_j are consistent ((J-1) for I_k).
- (J-2). From requirement $\mathsf{vars}(F) \subseteq V$ for every claim follows $\mathsf{vars}(F_i) \cap \mathsf{vars}(F_j) \subseteq V_i \cap V_j$. Furthermore, for $r \in \{i, j\}$ is $V_i \cap V_j \subseteq \mathsf{vars}(A_r)$ ((J-2) for I_k). Thus, also $\mathsf{vars}(F_i) \cap \mathsf{vars}(F_j) \subseteq \mathsf{vars}(A_r|_{\mathsf{vars}(F_r)})$.

The resulting claim is $(F_i \cup F_j, A_i|_{\mathsf{vars}(F_i)} \cup A_j|_{\mathsf{vars}(F_j)})$. We will show that

$$A_i|_{\mathsf{vars}(F_i)} \cup A_j|_{\mathsf{vars}(F_j)} = A_i|_{\mathsf{vars}(F_i)\cup\mathsf{vars}(F_j)} \cup A_j|_{\mathsf{vars}(F_i)\cup\mathsf{vars}(F_j)}.$$

The direction $\subseteq$ follows directly. For the other direction $\supseteq$ we assume that x is in the right set and show that x is in the left set as well. W.l.o.g. let $x \in A_i|_{\mathsf{vars}(F_i)\cup\mathsf{vars}(F_j)}$. If $x \in \mathsf{vars}(F_i)$, then $x \in A_i|_{\mathsf{vars}(F_i)}$. So assume $x \notin \mathsf{vars}(F_i)$, then is $x \in \mathsf{vars}(F_j)$. Our requirement $\mathsf{vars}(F_j) \subseteq V_j$ implies $x \in V_j$. Per definition of a claim, $\mathsf{vars}(A_i) \subseteq V_i$ and therefore, $x \in V_i$. Using (J-2) we get $x \in V_i \cap V_j \subseteq \mathsf{vars}(A_j)$. Since A_i and A_j are consistent ((J-1) for I_k), we have $x \in A_j|_{\mathsf{vars}(F_j)}$.

Therefore, the (Join') application results in the claim

$$\begin{aligned}&\big(F_i \cup F_j, A_i|_{\mathsf{vars}(F_i)\cup\mathsf{vars}(F_j)} \cup A_j|_{\mathsf{vars}(F_i)\cup\mathsf{vars}(F_j)}\big)\\ &\quad= \big(F_i \cup F_j, (A_i \cup A_j)|_{\mathsf{vars}(F_i)\cup\mathsf{vars}(F_j)}\big)\\ &\quad= f(I_k).\end{aligned}$$

Extension. $I_k = ((F, V), A)$ is derived with (Ext) from claim $I_j = ((F_j, V_j), A_j)$ with $j < k$ and $f(I_j) = (F_j, A_j|_{\mathsf{vars}(F_j)})$. We can derive $f(I_k) = (F, A|_{\mathsf{vars}(F)})$ with (Ext') of $f(I_j)$:

- (E-1). $F_j \subseteq F$ follows from (E-1) for I_k.
- (E-2). $(A|_{\mathsf{vars}(F)})|_{\mathsf{vars}(F_j)} = A|_{\mathsf{vars}(F)\cap\mathsf{vars}(F_j)} = A|_{\mathsf{vars}(F_j)}$ since $F_j \subseteq F$. Using $\mathsf{vars}(F_j) \subseteq V_j$ this is equal to $A|_{V_j\cap\mathsf{vars}(F_j)}$ which we can transform to $(A|_{V_j})|_{\mathsf{vars}(F_j)}$. Finally we can use $A|_{V_j} = A_j$ ((E-3) for I_k) and get $(A_j)|_{\mathsf{vars}(F_j)}$.
- (E-3). $A|_{\mathsf{vars}(F)}$ satisfies $F \setminus F_j$ since A satisfies $F \setminus F_j$ ((E-4) for I_k).

This completes the induction. Therefore, π' is a valid MICE' trace. Since the final claim is $f(I_m) = f(((\varphi, \mathsf{vars}(\varphi)), \emptyset)) = (\varphi, \emptyset)$ we have that π' is a MICE' proof of φ. Per construction, $c(\pi') \leqslant c(\pi) + 1$ and $s(\pi') \leqslant s(\pi) + 1$. The additional 1 is needed in order to use the (SA) rule to simulate (1-Mod). Apart from that, the number of claims and the number of clauses in the resolution refutations do not increase. ☐

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 4.9. MICE *p-simulates* MICE'.

Proof: 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 $I'_i = ((F_i, \mathsf{vars}(F_i)), A_i)$ and show that we can derive I'_k using $I'_1, \ldots I'_{k-1}$ with $O(|\mathsf{vars}(\varphi)|)$ MICE steps. We distinguish how I_k is derived.
Axiom. $I_k = (\emptyset, \emptyset)$ is derived with (Ax). Then we can derive $I'_k = ((\emptyset, \emptyset), \emptyset)$ with (1-Mod).

(O-1) and (O-2) are fulfilled since $\mathsf{vars}(\emptyset) = \emptyset$ and the empty assignment satisfies $\emptyset$.
Composition. $I_k = (F_k, A_k)$ is derived with (Comp') using absence of models statement ρ and claims $I_{i_1}, \ldots, I_{i_r}$ with $I_{i_j} = (F_k, A_{i_j})$ for $i_j < k$. Then we can derive I'_k with (Comp') from $I'_{i_1}, \ldots, I'_{i_r}$.

(C-1) and (C-2) follow directly from (C-1) and (C-2) for I_k as we do not modify the assumptions. For (C-3) we can simply use the absence of models statement ρ since it refutes

$$(A_k)|_{\mathsf{vars}(F_k)} \cup F_k \cup \{\overline{A}_{i_j}|_{\mathsf{vars}(F_k)} \mid j \in [r]\} = A_k \cup F_k \cup \{\overline{A}_{i_j} \mid j \in [r]\}.$$

Join. $I_k = (F_i \cup F_j, A_i \cup A_j)$ is derived with $(\mathsf{Join'})$ applied to $I_i = (F_i, A_i)$ and $I_j = (F_j, A_j)$ with $i, j < k$. Then we can derive I'_k with $(\mathsf{Join'})$ using I'_i and I'_j.

(J-1) follows directly from (J-1) for I_k, as we do not modify the assumptions. (J-2) stating $\mathsf{vars}(F_1) \cap \mathsf{vars}(F_2) \subseteq \mathsf{vars}(A_k)$ follows from (J-2) for I_k.

Extension. $I_k = (F_k, A_k)$ is derived with $(\mathsf{Ext'})$ from $I_i = (F_i, A_i)$ with $i < k$.

We derive

$$I = ((\emptyset, \mathsf{vars}(F_k) \setminus (\mathsf{vars}(F_i) \cup \mathsf{vars}(A_k))), \emptyset)$$

with the construction of Proposition 3.3. We can apply (Join) to I and I'_i.

- (J-1). The empty assumption $\emptyset$ and A_i are consistent.
- (J-2). $(\mathsf{vars}(F_k) \setminus (\mathsf{vars}(F_i) \cup \mathsf{vars}(A_k))) \cap \mathsf{vars}(F_i) = \emptyset \subseteq \mathsf{vars}(A_j)$.
- (J-3). This follows from Corollary 3.7.

With this $(\mathsf{Join'})$ we receive

$$\begin{aligned} I' &= ((F_i, \mathsf{vars}(F_i) \cup \mathsf{vars}(F_k) \setminus (\mathsf{vars}(F_i) \cup \mathsf{vars}(A_k))), A_i) \\ &= ((F_i, \mathsf{vars}(F_i) \cup \mathsf{vars}(F_k) \setminus \mathsf{vars}(A_k)), A_i). \end{aligned}$$

Next, we can apply (Ext) to get

$$((F_k, \mathsf{vars}(F_k)), A_k) = I'_k.$$

- (E-1). $F_i \subseteq F_k$ follows from (E-1) for I_k. Therefore, we also have $\mathsf{vars}(F_i) \cup \mathsf{vars}(F_k) \setminus \mathsf{vars}(A_k) \subseteq \mathsf{vars}(F_k)$.
- (E-2). We apply some basic set operations to get $\mathsf{vars}(F_k) \setminus (\mathsf{vars}(F_i) \cup (\mathsf{vars}(F_k) \setminus \mathsf{vars}(A_k))) \subseteq \mathsf{vars}(F_k) \setminus (\mathsf{vars}(F_k) \setminus \mathsf{vars}(A_k)) \subseteq \mathsf{vars}(F_k) \cap \mathsf{vars}(A_k) = \mathsf{vars}(A_k)$. For the last equation we used that (F_k, A_k) is a $\mathsf{MICE'}$ claim and therefore $\mathsf{vars}(A_k) \subseteq \mathsf{vars}(F_k)$.
- (E-3). We have that $A_k|_{\mathsf{vars}(F_i) \cup \mathsf{vars}(F_k) \setminus \mathsf{vars}(A_k)} = A_k|_{\mathsf{vars}(F_i)}$ is equal to A_i because of (E-2) for I_k.
- (E-4). A_k satisfies $F_k \setminus F_i$ follows from (E-3) for I_k.
- (E-5). This follows from Corollary 3.7.

As a result, we can derive I'_k from $I'_1, \ldots I'_{k-1}$ with a single MICE step if I_k is derived with (Ax), $(\mathsf{Comp'})$ or $(\mathsf{Join'})$. In particular, the resolution proof size of the absence of models statement in case of $(\mathsf{Comp'})$ does not change. If I'_k is derived with $(\mathsf{Ext'})$, we need one application of the construction of Proposition 3.3, one (Join) and one (Ext) and therefore in total $O(|\mathsf{vars}(\varphi)|)$ MICE steps.

Since $I'_n = ((\varphi, \mathsf{vars}(\varphi)), \emptyset)$, there is a MICE proof π' of φ that has sizes $s(\pi') = s(\pi) \cdot O(\mathsf{vars}(\varphi))$ and $c(\pi') = c(\pi) \cdot O(\mathsf{vars}(\varphi))$. ☐

5. Lower Bounds for MICE and MICE'

In this section we investigate the proof complexity of $\mathsf{MICE'}$. Because of the equivalence of MICE and $\mathsf{MICE'}$ (Proposition 4.8 and Proposition 4.9), all of the proof complexity results for $\mathsf{MICE'}$ below also apply to 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 5.1. MICE' *is polynomially equivalent to* Res *for unsatisfiable formulas.*

Proof: 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 $I_i = (F_i, A_i, c_i)$. We define $\pi_{\mathsf{Res}} = (\varphi, X_1, X_2, \ldots, X_m)$ with

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

We show that π_{Res} is a valid resolution trace (with weakening steps). For this we use induction on m. In the base case for $m = 0$ the trace only contains the clauses of φ and is therefore valid. For the induction step let $(\varphi, X_1, \ldots, X_{k-1})$ be a valid resolution trace. If $c_k \neq 0$, there is nothing to show. Therefore, we can assume that $c_k = 0$. In particular, I_k is not derived with (Ax). We distinguish how I_k is derived.

- I_k is derived with (Comp') from claims $I_{i_1}, \ldots, I_{i_r}$ with $i_j < k$ using the absence of models statement ρ which is a resolution derivation

  $$F_k \cup A_k \cup \{\overline{A_{i_j}} \mid j \in [r]\} \vdash \Box.$$

  In this derivation we can weaken every clause by $\overline{A_k}$. Thus X_k is a resolution derivation of

  $$F_k \cup \{\overline{A_{i_j}} \mid j \in [r]\} \vdash \overline{A_k}.$$

  All clauses of $F_k \subseteq \varphi$ (Observation 3.4) are already in π_{Res}. All clauses $\overline{A_{i_j}}$ are in π_{Res} as well by induction hypothesis, since $c_{i_j} = 0$ for all used claims $I_{i_1}, \ldots, I_{i_r}$ to get the sum $c_k = 0$. Thus, the resolution derivation is correct.

- I_k is derived by (Join') from claims I_i and I_j with $i, j < k$. Since $c_k = 0 = c_i \cdot c_j$, w.l.o.g. $c_i = 0$. Therefore, we have already derived $\overline{A}_i$ by induction hypothesis. Thus $\overline{A}_k = \overline{A_i \cup A_j} = (\overline{A}_i \vee \overline{A}_j)$ can be derived with a single weakening step.
- I_k is derived by (Ext') from I_i with $i < k$. Since $c_k = c_i \cdot 2^{|\mathsf{vars}(F_k) \setminus (\mathsf{vars}(F_i) \cup \mathsf{vars}(A_k))|} = 0$ we have $c_i = 0$. Thus $\overline{A}_i$ has already been derived. We can derive $\overline{A}_k$ from $\overline{A}_i$ by weakening since $A_i \subseteq A_k$ (E-3).

Since $c_m = 0$, the last claim I_m is derived with (Comp'), (Join') or (Ext'). Thus, X_m contains the clause $\overline{A}_m = \Box$. As a result, π_{Res} is a resolution refutation of φ since it is a valid derivation of $\Box$. Furthermore, we see that $|\pi_{\mathsf{Res}}| = O(s(\pi))$. It is known that any refutation with resolution and weakening can be transformed into a refutation without weakening efficiently which proves the claim. ☐

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 [27]. With Theorem 5.1 we can conclude that these hard formulas for resolution are also hard for MICE′.

Corollary 5.2. *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 5.1) 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 5.3. *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 4.6 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 5.4. *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 $c(\pi') \leqslant c(\pi)$.*

Proof: Assume that in π some claim $I = (F, A)$ is derived with (Comp') from claims $I_1, \dots, I_n$ with $I_i = (F, A_i, c_i)$ and $c_n = 0$ using some absence of models statement ρ. Because of Theorem 5.1 we can construct a resolution refutation ρ_n of $F \cup A_n$ that has size $O(|\pi|)$. Therefore, we can derive I with (Comp') from $I_1, \dots, I_{n-1}$ as well: (C-1) and (C-2) are still satisfied. For (C-3) we need an absence of models statement that refutes $F \cup A \cup \{\overline{A}_i \mid i \in [n-1]\}$. For this we can first derive $\overline{A}_n$ from F with ρ_n and then apply ρ. Like this, we can remove claim I_n. We repeat this for every claim with $c = 0$ that is used for (Comp'). Afterwards, we remove all claims that became redundant. Let π' be the resulting proof.

Per construction, π' is a valid MICE' proof for φ. We will show that π' has no claims with $c = 0$. Assume otherwise claim I with $c = 0$ is in π'. Since I is not redundant, there is a path to the final claim with $c > 0$. In this path there have to be claims I_1 with $c_1 = 0$ and I_2 with $c_2 > 0$ such that I_2 is directly derived from I_1 with one of the four MICE' rules.

- Obviously this is not possible with (Ax).
- Per construction, it is impossible with (Comp'), because otherwise I_1 would not be in π'.
- It is not possible with (Join') nor (Ext') as c_2 would be a product with one factor $c_1 = 0$ leading to $c_2 = 0$.

Hence, π' has no claim with $c = 0$. Furthermore, $c(\pi') \leqslant c(\pi)$ since we only removed claims. For every claim with $c = 0$ that was used for (Comp'), we have to add a resolution proof of size $O(s(\pi))$ leading to $s(\pi') = O(s(\pi)^2)$. □

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 5.5. The formula $\mathsf{XOR\text{-}PAIRS}_n$ consists of the clauses

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

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

Theorem 5.6. *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 5.7. $\mathsf{XOR\text{-}PAIRS}_n$ *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 $\mathsf{XOR\text{-}PAIRS}_n$. $\square$

For the following arguments we will only consider MICE' proofs of $\mathsf{XOR\text{-}PAIRS}_n$ without redundant claims (i.e. all claims are connected to the final claim) and without claims with $c = 0$ (this is possible by Lemma 5.4). 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 5.8. *Let (F, A) be an arbitrary claim in a MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$. 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^k_{ij} for $k \in [4]$ that is not in F.*

Proof: We fix variables $i, j \in [n]$ such that $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F)$ and a clause $C = C^k_{ij} \notin F$ for some $k \in [4]$. We consider only the path from (F, A) to $(\mathsf{XOR\text{-}PAIRS}_n, \emptyset)$ 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.

- I_2 is obviously not derived with (Ax) nor (Comp'), since $F_1 \neq F_2$.
- Assume I_2 is derived with (Join') of I_1 and some $I_3 = (F_3, A_3)$. Since $C \notin F_1$ and $C \in F_2 = F_1 \cup F_3$ is $C \in F_3$. In particular $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F_3)$. Together with $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F) \subseteq \mathsf{vars}(F_1)$ we get $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(F_1) \cap \mathsf{vars}(F_3) \subseteq \mathsf{vars}(A_1)$ and $\{x_i, x_j, z_{ij}\} \subseteq \mathsf{vars}(A_3)$ where we used (J-2). Since A_1 and A_3 are consistent (J-1), x_i, x_j, z_{ij} have to be assigned in the same way in A_1 and A_3. Because of Lemma 4.7 those variables have to be set in A as well and in particular with the same polarities. Assume A does not satisfy C. Then, A_3 does not satisfy C either, since all variables of C are set as in A. Hence, (F_3, A_3) has no models leading to $c_3 = 0$ which contradicts our assumption that there are no claims with count zero for satisfiable formulas (Lemma 5.4). Therefore, A has to satisfy C.
- Assume, I_2 is derived with (Ext') from I_1. Then A_2 has to satisfy $C \in F_2 \setminus F_1$ by condition (E-3). Because of Lemma 4.7, A has to assign x_i, x_j, z_{ij} in the same way as A_2. Hence A satisfies C as well.

Therefore, I_2 can only be derived if A satisfies C leading to the lemma. $\square$

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 5.9. *Let a MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$ 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: We fix indices $i, j \in [n]$ such that $\{x_i, x_j\} \subseteq \mathsf{vars}(F)$ and $z_{ij} \notin \mathsf{vars}(F)$. Since $z_{ij} \notin \mathsf{vars}(F)$ we have $C^k_{ij} \notin F$ for all $k \in [4]$. We consider only the path from (F, A) to $(\mathsf{XOR\text{-}PAIRS}_n, \emptyset)$ 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 $C^k_{ij} \notin F_1$ for all $k \in [4]$ and $C^s_{ij} \in F_2$ for at least one $s \in [4]$. That means, I_1 is the last claim in this path which contains none of the four clauses C_{ij}. Towards a contradiction, let us assume $x_i \notin \mathsf{vars}(A)$

(the argument for x_j is analogous). By Lemma 4.7, also $x_i \notin \mathsf{vars}(A_1)$ and $x_i \notin \mathsf{vars}(A_2)$. The claim I_2 is directly derived from I_1 by one of the four MICE' rules.

- I_2 is not derived with (Ax) nor (Comp'), since $F_1 \neq F_2$.
- I_2 is not derived with (Join'). Assume otherwise, then I_2 is the result of (Join') of I_1 with some other claim $I_3 = (F_3, A_3)$. Since $C^s_{ij} \notin F_1$ and $C^s_{ij} \in F_2 = F_1 \cup F_3$ we have $C^s_{ij} \in F_3$ and in particular $x_i \in \mathsf{vars}(F_3)$. Together with $x_i \in \mathsf{vars}(F) \subseteq \mathsf{vars}(F_1)$ we get a contradiction with (J-2) because $x_i \in \mathsf{vars}(F_1) \cap \mathsf{vars}(F_3) \subseteq \mathsf{vars}(A_1)$, but $x_i \notin \mathsf{vars}(A_1)$.
- I_2 is not derived with (Ext') from I_1. Otherwise, A_2 has to satisfy $F_2 \backslash F_1$ by condition (E-3). Since F_1 does not contain any clause C_{ij}, A_2 has to satisfy all clauses C_{ij} that are in F_2. By Lemma 5.8, A_2 has to satisfy all clauses C_{ij} that are not in F_2 as well. In order to satisfy all four clauses of C_{ij}, all three variables x_i, x_j and z_{ij} have to be set in A_2, in particular $x_i \in \mathsf{vars}(A_2)$ which is a contradiction.

As a result, I_2 cannot be derived from I_1 which implies that our assumption $x_i \notin \mathsf{vars}(A)$ was false. □

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 $\mathsf{XOR\text{-}PAIRS}$ formulas.

Lemma 5.10. *Let a MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$ 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, $c_1 \geqslant 2$ and $c_2 \geqslant 2$.

Assume that all x variables occurring in $\mathsf{vars}(F_1)$ are assigned in A_1. Since $c_1 \geqslant 2$, $\mathsf{vars}(F_1) \setminus \mathsf{vars}(A_1) \neq \emptyset$. 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^k_{ij} that are in F_1. Because of Lemma 5.8, A_1 has to satisfy the clauses C^k_{ij} that are not in F_1 as well. Thus, A_1 has to satisfy all four clauses C^k_{ij}, 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 $C^k_{ij} \notin F$ for all k, implying $z_{ij} \notin \mathsf{vars}(F)$. With Lemma 5.9 we get the contradiction $x_j \in \mathsf{vars}(A) = \mathsf{vars}(A_1 \cup A_2)$.

Therefore, our assumption $c_1 \geqslant 2$ and $c_2 \geqslant 2$ was false. □

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

Lemma 5.11. *Let π be a MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$. Then there is a MICE' proof π' that does not use (Join') with $c(\pi') \leqslant 2 \cdot c(\pi)$.*

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 5.10 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 $\mathsf{XOR\text{-}PAIRS}_n$ that does not use (Join'). The number of claims in the proof increases at most by a factor of two. □

Lemma 5.12. *Let a MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$ 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 5.8. 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 $\mathsf{XOR\text{-}PAIRS}$ formulas require proofs with an exponential number of MICE' steps.

Proof of Theorem 5.6: Note that with Observation 5.7, Lemma 5.10 and Lemma 5.12 we can infer immediately that any tree-like MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$, i.e. any proof that uses every claim except the axiom at most one time, has at least size $2^n + 2$. However, in general (dag-like) MICE' proofs, any claim can be used multiple times. General dag-like MICE' might be exponentially stronger than the tree-like version. 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 $\mathsf{XOR\text{-}PAIRS}_n$. Let π' be a MICE' proof of $\mathsf{XOR\text{-}PAIRS}_n$ that does not use (Join') with $c(\pi') \leqslant 2 \cdot c(\pi)$ which has to exist because of Lemma 5.11.

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 5.12, 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 5.8. 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 4.7 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 4.3 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 $c(\pi) \geqslant \frac{1}{2} \cdot c(\pi') \geqslant 2^{n-1}$ leading to the lower bound. □

6. Connection Between MICE' and Decision DNNFs

In this section we show that there is a tight connection between MICE' proofs and decision DNNFs. That is, we show in Section 6.1. that we can extract a decision DNNF for some formula φ efficiently from a MICE' proof of φ. By exploiting this connection we immediately get further lower bounds for MICE'. Finally, in Section 6.2. we use this connection in order to provide an alternative proof that $\mathsf{XOR\text{-}PAIRS}$ requires MICE' proofs of exponential size.

Let us first review the concept of DNNFs (Decomposable Negation Normal Form) and focus on the special case of *decision* DNNFs [21] which are widely used in knowledge compilation. Formally, a decision DNNF D with variables V is a directed acyclic graph with the following conditions. It has exactly one node with in-degree 0 which is called source. The nodes with outdegree 0 are labelled with 0 or 1 and are called sinks. All other nodes have out-degree 2 and are either *decision* nodes or And nodes. A *decision node* is labelled with a variable $x \in V$. One outgoing edge is labelled with 0 and the other with 1. On any path in D the variable x can be decided at most once. An And node is labelled with $\wedge$ and has to satisfy the decomposability property. That is, the sets of variables that occur in the subcircuits of the two children have to be disjoint.

Let N be any node in D, then D_N denotes the subcircuit of D with root N. Under a given assignment $\alpha \in \langle V \rangle$, D evaluates to $D(\alpha)$ which is defined recursively as follows.

- Let N be a sink of D with label 0, then $D_N(\alpha) = 0$. If its label is 1, then $D_N(\alpha) = 1$.
- Let N be a decision node deciding variable $x \in V$ and let N_0 be the child node for $x = 0$, N_1 for $x = 1$. Then,

$$D_N(\alpha) = \begin{cases} D_{N_0}(\alpha), & \text{if } x \text{ is assigned to false in } \alpha \\ D_{N_1}(\alpha), & \text{if } x \text{ is assigned to true in } \alpha. \end{cases}$$

- Let N be an And node with children N_0 and N_1. Then, $D_N(\alpha) = D_{N_0}(\alpha) \wedge D_{N_1}(\alpha)$.

A decision DNNF represents some formula φ if D evaluates to 1 for exactly the models of φ. The size of a decision DNNF is the number of its nodes.

6.1. Efficient Extraction of a Decision DNNF from a MICE' Proof

We will show, that we can extract decision DNNFs from MICE' proofs efficiently, i.e. the size of the resulting decision DNNF is not much larger than the number of MICE' steps.

Theorem 6.1. *Let φ be a formula with MICE' proof π with n steps. Then there exists a decision DNNF of size at most $n \cdot (1 + |\mathsf{vars}(\varphi)|) + 1$ representing φ.*

Proof: Let $\pi = I_1, \ldots, I_n$ be a MICE' proof with $I_k = (F_k, A_k)$ for every $k \in [n]$. Our goal is to construct from π a decision DNNF for φ. W.l.o.g. the first claim of π is $(\emptyset, \emptyset, 1)$ derived with (Ax) and all other claims are not derived with (Ax). We use the notation $\mathsf{Mod}_\varphi(F) := \{\alpha \in \langle \mathsf{vars}(\varphi) \rangle \mid \alpha \models F\}$. Inductively, we construct a decision DNNF C_k for every $k \in [n]$ such that:

- (IH1) C_k evaluates to one on exactly all assignments from $\mathsf{Mod}_\varphi(F_k[A_k])$ and
- (IH2) C_k contains only variables from $F_k[A_k]$.

For the base case $k = 1$, $I_1 = (\emptyset, \emptyset, 1)$ is derived with (Ax). Therefore, we set C_1 to a circuit that only contains one sink labelled with 1.

For the induction step we distinguish how I_k is derived.

Join. I_k is derived with (Join) of claims I_i and I_j. Per induction hypothesis, we have already derived decision DNNFs C_i and C_j representing $\mathsf{Mod}_\varphi(F_i[A_i])$ and $\mathsf{Mod}_\varphi(F_j[A_j])$. We define C_k to be an And gate with the two children C_i and C_j. Because of (J-2) we have

$\mathsf{vars}(F_i) \cap \mathsf{vars}(F_j) \subseteq \mathsf{vars}(A_i) \cap \mathsf{vars}(A_j)$. Together with (IH2) we get $\mathsf{vars}(C_i) \cap \mathsf{vars}(C_j) \subseteq \mathsf{vars}(F_i[A_i]) \cap \mathsf{vars}(F_j[A_j]) = \emptyset$. Therefore, the And is indeed decomposable. Furthermore, (IH1) and (IH2) are satisfied:

$$\begin{aligned}
&\mathsf{Mod}_\varphi(F_k[A_k])\\
&\quad = \mathsf{Mod}_\varphi((F_i \cup F_j)[A_i \cup A_j])\\
&\quad = \mathsf{Mod}_\varphi(F_i[A_i \cup A_j] \cup F_j[A_i \cup A_j])\\
&\quad = \mathsf{Mod}_\varphi(F_i[A_i] \cup F_j[A_j])\\
&\quad = \mathsf{Mod}_\varphi(F_i[A_i]) \cap \mathsf{Mod}_\varphi(F_j[A_j])
\end{aligned}$$

and

$$\begin{aligned}
\mathsf{vars}(C_k) &= \mathsf{vars}(C_i) \cup \mathsf{vars}(C_j)\\
&\subseteq \mathsf{vars}(F_i[A_i]) \cup \mathsf{vars}(F_j[A_j])\\
&\subseteq \mathsf{vars}((F_i \cup F_j)[A_i \cup A_j])\\
&= \mathsf{vars}(F_k[A_k])
\end{aligned}$$

where we used that A_i, A_j are consistent (J-1). Further, in the third step, we use that if there is some variable $v \in \mathsf{vars}(F_i) \cap \mathsf{vars}(A_j)$, then also $v \in \mathsf{vars}(F_i) \cap \mathsf{vars}(F_j) \subseteq \mathsf{vars}(A_i)$ (J-2).

Composition. I_k is derived with (Comp) from claims $I_{i_1}, \ldots, I_{i_r}$. If $r = 0$, then C_k only contains one node labelled with 0 and the induction hypothesis is fulfilled. Otherwise, let $V = \mathsf{vars}(A_{i_1}) \setminus \mathsf{vars}(A_k)$. (Remember, that all assumptions A_{i_j} have the same set of variables because of (C-1)). We build a complete binary decision tree T with variables in V. For every claim I_{i_j} for $j \in [r]$ there is exactly one leaf in T that is consistent with the assumption of I_{i_j}. We replace this leaf with the corresponding decision DNNF C_{i_j}. Afterwards, we replace all remaining leaves with the 0 sink. Furthermore, we remove every decision gate where both decisions lead to the 0 sink node as long as such nodes exist. We set C_k to be the resulting circuit. Note, that C_k has at most n paths from the root to some claim and every such path has at most $|\mathsf{vars}(\varphi)|$ decision nodes.

Per construction, C_k contains exactly the models of $F_k[A_k]$ and C_k contains only variables from $F_k[A_k]$.

Extension. I_k is derived with (Ext) from I_i. Then we can set $C_k = C_i$. To see this, we use that A_k satisfies $F_k \setminus F_i$ by (E-3) and $A_k|_{\mathsf{vars}(F_i)} = A_i$ by (E-2):

$$\begin{aligned}
F_k[A_k] &= F_i[A_k]\\
&= F_i[A_k|_{\mathsf{vars}(F_i)}]\\
&= F_i[A_i].
\end{aligned}$$

Therefore, $\mathsf{Mod}_\varphi(F_k[A_k]) = \mathsf{Mod}_\varphi(F_i[A_i])$.

This completes the induction. Since $I_n = (\varphi, \emptyset)$, C_n computes $\mathsf{Mod}_\varphi(\varphi)$. Thus, C_n represents φ and is a decision DNNF by construction.

To estimate the size, we observe that every claim becomes a node in C_n. Further, there are at most $|\mathsf{vars}(\varphi)| \cdot n$ additional decision nodes for the (Comp') constructions. We may also need one additional sink labelled with 0. In total, we get $|C_n| \leqslant n \cdot (1 + |\mathsf{vars}(\varphi)|) + 1$. $\square$

As a direct consequence of Theorem 6.1, lower bounds for decision DNNFs also apply for MICE'.

Corollary 6.2. *If formula φ requires a decision DNNF of size d, then any MICE' proof for φ has size at least $(d-1)\cdot(|\mathsf{vars}(\varphi)|+1)^{-1}$.*

However, the resulting lower bounds are only of interest if the formulas admit short CNF representations. Therefore, in order to obtain relevant lower bounds for MICE', we need formulas that separate CNF from decision DNNFs. In fact, a few such formulas are known in the literature [4,5,11,12], which by Corollary 6.2 yield additional MICE' lower bounds.

6.2. An Alternative Proof of the Lower Bound

Here, we provide an alternative proof to our direct MICE' lower bound for $\mathsf{XOR\text{-}PAIRS}$ (Theorem 5.6). By Corollary 6.2, to show the lower bound it suffices to show that $\mathsf{XOR\text{-}PAIRS}$ require decision DNNFs of exponential size. In fact, we show the even stronger result, that all DNNFs for $\mathsf{XOR\text{-}PAIRS}$ have exponential size.

For the DNNF size lower bound we use a technique from communication complexity similar to [12]. To do so, we have to introduce some notions from communication complexity. Let V be a set of variables. A *(combinatorial) rectangle* over V is a set $R\subseteq\langle V\rangle$ such that there exists a partition $V=V_1\uplus V_2$ and two sets of assignments $r_1\subseteq\langle V_1\rangle$, $r_2\subseteq\langle V_2\rangle$ satisfying $R=\{\alpha_1\cup\alpha_2 \mid \alpha_1\in r_1,\alpha_2\in r_2\}$. A rectangle is called *balanced* if its underlying partition is balanced, i.e. $\frac{|V|}{3}\leqslant|V_1|\leqslant\frac{2\cdot|V|}{3}$. A finite set of balanced rectangles $\{R_i\}$ over variables $\mathsf{vars}(\varphi)$ is called *rectangle cover* of φ if $\bigcup_i R_i=\mathsf{Mod}_\varphi$.

The following result provides a powerful technique to prove lower bounds for DNNF size (and thus also for MICE' proof size).

Theorem 6.3 ([12])**.** *Let C be a DNNF computing a function φ. Then, φ has a balanced rectangle cover of size at most $|C|$.*

Therefore, we only have to prove that any rectangle cover of $\mathsf{XOR\text{-}PAIRS}$ has exponential size. For that, we show that any rectangle in such a cover cannot be too large.

Lemma 6.4. *Any balanced rectangle for $\mathsf{XOR\text{-}PAIRS}_n$ has size at most $2^{\frac{73}{74}\cdot n}$ for large enough n.*

Proof: Let n be large enough and R be a balanced rectangle from some arbitrary rectangle cover for $\mathsf{XOR\text{-}PAIRS}_n$. Let $V=V_1\uplus V_2$ be the underlying balanced partition. We say that a pair (i,j) is *split* if x_i, x_j and $z_{i,j}$ do not all occur in the same set V_1 or V_2. Further, two pairs (i,j) and (k,l) *intersect* if $\{i,j\}\cap\{k,l\}\neq\emptyset$.

First, we show that R contains at least $\frac{n^2}{37}$ pairs that are split. For that we distinguish two cases.

Case 1. Assume that both sets V_1 and V_2 contain at least $\frac{n}{6}$ different x_i variables each. Then (i,j) is split, if $x_i\in V_1$ and $x_j\in V_2$. Thus, R has at least $(\frac{n}{6})^2=\frac{n^2}{36}$ split pairs.

Case 2. Otherwise we assume that V_2 has w.l.o.g. at most $\frac{n}{6}$ different x_i variables. Since V_1 has at least $\frac{5\cdot n}{6}$ different x_i variables, there are $(\frac{5\cdot n}{6})^2=\frac{25\cdot n^2}{36}$ different z_{ij} variables that would need to be in V_1 such that V_1 does not contain a split pair. However, as R is balanced, we have $|V_1| \leqslant \frac{2}{3} \cdot |V| = \frac{2}{3} \cdot (n^2 + n)$. Therefore, these z_{ij} variables do not all fit in V_1 and for every such variable z_{ij} that is put in V_2 instead, we obtain a split pair (i, j). In this way, by making V_1 as large as allowed, we still get $\frac{25 \cdot n^2}{36} - \frac{2}{3} \cdot (n^2 + n) = \frac{n^2}{36} - \frac{2 \cdot n}{3} \geqslant \frac{n^2}{37}$ (for n large enough) split pairs.

Next, we have a closer look at these $\frac{n^2}{37}$ split pairs. Since every pair (i, j) only intersects with at most $2 \cdot n$ other pairs, R has to contain at least $\frac{n^2/37}{2 \cdot n} = \frac{n}{74}$ split pairs that are pairwise-disjoint.

Let (i, j) be one of these pairwise disjoint pairs. For the two variables x_i and x_j there are four possibilities to assign them. We can show that at most two of these assignments lie in our rectangle R. For that we distinguish two cases.

Case 1. x_i and x_j are in two different sets of V_1 and V_2. W.l.o.g. we assume that $x_i \in V_1$, $x_j \in V_2$ and $z_{ij} \in V_2$. Then, the value of x_i has to be fixed in R. Otherwise, R would contain two assignments that assign x_j and z_{ij} in the same way but x_i differently. As R contains only assignments that satisfy $\mathsf{XOR\text{-}PAIRS}_n$ this is impossible because it contradicts $z_{ij} = x_i \oplus x_j$.

Case 2. x_i and x_j are in the same set. W.l.o.g. we assume that $x_i \in V_1$ and $x_j \in V_1$. Since (i, j) is split, we have $z_{ij} \in V_2$. With the same argument used in case 1, we see that the value of $x_i \oplus x_j$ has to be constant in R. So there are at most two of the four assignments for x_1 and x_2 in R.

Now, we can finally argue about the maximum size R can have. As $\mathsf{XOR\text{-}PAIRS}_n$ has 2^n models, it cannot be larger than that. However, for every pairwise-disjoint pair in R, the rectangle can only contain two of the four possible assignments. Therefore, $|R| \leqslant 2^{n - \frac{n}{74}} = 2^{\frac{73}{74} \cdot n}$. □

For $\mathsf{XOR\text{-}PAIRS}_n$ we have to cover all 2^n models with rectangles which cannot be larger than $2^{\frac{73}{74} \cdot n}$. Therefore, any cover has at least size $2^{\frac{n}{74}}$. With Theorem 6.3, we obtain the DNNF size lower bound.

Corollary 6.5. *Any DNNF computing* $\mathsf{XOR\text{-}PAIRS}_n$ *has size at least* $2^{\frac{n}{74}}$ *for large enough* n.

Finally, by applying Theorem 6.1, we get the MICE' lower bound as well.

Corollary 6.6. *Any* MICE' *proof of* $\mathsf{XOR\text{-}PAIRS}_n$ *has size* $2^{\Omega(n)}$.

7. 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. [24] show that MICE proofs can be extracted from solver runs for sharpSAT [35], DPDB [25] and D4 [30], 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.

It would also be interesting to determine the exact relations between the systems MICE, MICE′ and the two other #SAT proof systems kcps(#SAT) [16], based on certified decision DNNFs, and CPOG [13].

Acknowledgements

An extended abstract of this work appeared in the proceedings of SAT'23 [7]. We thank an anonymous reviewer of SAT'23 for pointing out the connection between MICE' and decision DNNFs resulting in the alternative approach to obtain lower bounds for MICE' described in Section 6. We also thank an anonymous reviewer of this journal article for providing a proof sketch for the alternative lower bound argument in Section 6.2 which they kindly allowed us to reproduce here.

Research was supported by a grant from the Carl Zeiss Foundation and DFG grant BE 4209/3-1.

References

[1] A. Atserias, J.K. Fichte and M. Thurley, Clause-learning algorithms with many restarts and bounded-width resolution, *J. Artif. Intell. Res.* **40** (2011), 353–373. doi:10.1613/jair.3152.

[2] F. Bacchus, S. Dalmao and T. Pitassi, Algorithms and complexity results for #SAT and Bayesian inference, in: *44th Symposium on Foundations of Computer Science (FOCS 2003), Proceedings*, Cambridge, MA, USA, 11–14 October 2003, IEEE Computer Society, 2003, pp. 340–351.

[3] T. Baluta, Z.L. Chua, K.S. Meel and P. 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, IEEE, 2021, pp. 312–323.

[4] P. Beame, J. Li, S. Roy and D. Suciu, Lower bounds for exact model counting and applications in probabilistic databases, in: *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013*, Bellevue, WA, USA, August 11–15, 2013, A.E. Nicholson and P. Smyth, eds, AUAI Press, 2013.

[5] P. Beame, J. Li, S. Roy and D. Suciu, Lower bounds for exact model counting and applications in probabilistic databases, in: *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013*, Bellevue, WA, USA, August 11–15, 2013, A.E. Nicholson and P. Smyth, eds, AUAI Press, 2013.

[6] O. Beyersdorff and B. Böhm, Understanding the relative strength of QBF CDCL solvers and QBF resolution, in: *12th Innovations in Theoretical Computer Science Conference (ITCS 2021), Leibniz International Proceedings in Informatics (LIPIcs)*, Vol. 185, 2021, pp. 12:1–12:20.

[7] O. Beyersdorff, T. Hoffmann and L.N. Spachmann, Proof complexity of propositional model counting, in: *26th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, M. Mahajan and F. Slivovsky, eds, LIPIcs, Vol. 271, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, pp. 2:1–2:18.

[8] A. Biere, M. Heule, H. van Maaren and T. Walsh (eds), *Handbook of Satisfiability, in Frontiers in Artificial Intelligence and Applications*, IOS Press, 2021.

[9] B. Böhm and O. Beyersdorff, Lower bounds for QCDCL via formula gauge, in: *Theory and Applications of Satisfiability Testing (SAT)*, C.-M. Li and F. Manyà, eds, Springer International Publishing, Cham, 2021, pp. 47–63.

[10] B. Böhm, T. Peitl and O. Beyersdorff, QCDCL with cube learning or pure literal elimination – what is best? in: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI)*, L.D. Raedt, ed., ijcai.org, 2022, pp. 1781–1787.

[11] S. Bova, F. Capelli, S. Mengel and F. Slivovsky, Expander CNFs have Exponential DNNF Size, 2014, CoRR, http://arxiv.org/abs/1411.1995, arXiv:1411.1995.

[12] S. Bova, F. Capelli, S. Mengel and F. Slivovsky, Knowledge compilation meets communication complexity, in: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, S. Kambhampati, ed., IJCAI/AAAI Press, 2016, pp. 1008–1014.

[13] R.E. Bryant, W. Nawrocki, J. Avigad and M.J.H. Heule, Certified knowledge compilation with application to verified model counting, in: *26th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, M. Mahajan and F. Slivovsky, eds, LIPIcs, Vol. 271, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, pp. 6:1–6:20.

[14] S. Buss and J. Nordström, Proof complexity and SAT solving, in: *Handbook of Satisfiability*, A. Biere, M. Heule, H. van Maaren and T. Walsh, eds, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021, pp. 233–350.

[15] S. Buss and N. Thapen, DRAT proofs, propagation redundancy, and extended resolution, in: *Theory and Applications of Satisfiability Testing (SAT)*, M. Janota and I. Lynce, eds, Lecture Notes in Computer Science, Vol. 11628, Springer, 2019, pp. 71–89.

[16] F. Capelli, Knowledge compilation languages as proof systems, in: *Theory and Applications of Satisfiability Testing (SAT)*, M. Janota and I. Lynce, eds, Lecture Notes in Computer Science, Vol. 11628, Springer, 2019, pp. 90–99.

[17] F. Capelli, J. Lagniez and P. Marquis, Certifying top-down decision-DNNF compilers, in: *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI) 2021*, AAAI Press, 2021, pp. 6244–6253.

[18] L. Chew and M.J.H. Heule, Relating existing powerful proof systems for QBF, in: *25th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, K.S. Meel and O. Strichman, eds, LIPIcs, Vol. 236, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, pp. 10:1–10:22.

[19] S.A. Cook, The complexity of theorem proving procedures, in: *Proc. 3rd Annual ACM Symposium on Theory of Computing*, 1971, pp. 151–158.

[20] S.A. Cook and R.A. Reckhow, The relative efficiency of propositional proof systems, *J. Symb. Log.* **44**(1) (1979), 36–50. doi:10.2307/2273702.

[21] A. Darwiche, Decomposable negation normal form, *J. ACM* **48**(4) (2001), 608–647. doi:10.1145/502090.502091.

[22] L. Dueñas-Osorio, K.S. Meel, R. Paredes and M.Y. Vardi, Counting-based reliability estimation for power-transmission grids, in: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, California, USA, February 4-9, 2017, S. Singh and S. Markovitch, eds, AAAI Press, 2017, pp. 4488–4494.

[23] J.K. Fichte, M. Hecher and F. Hamiti, The model counting competition 2020, *ACM J. Exp. Algorithmics* **26** (2021), 13:1–13:26. doi:10.1145/3459080.

[24] J.K. Fichte, M. Hecher and V. Roland, Proofs for propositional model counting, in: *25th International Conference on Theory and Applications of Satisfiability Testing, SAT 2022*, Haifa, Israel, August 2–5, 2022, K.S. Meel and O. Strichman, eds, LIPIcs, Vol. 236, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, pp. 30:1–30:24.

[25] J.K. Fichte, M. Hecher, P. Thier and S. Woltran, Exploiting database management systems and treewidth for counting, in: *Practical Aspects of Declarative Languages – 22nd International Symposium, PADL 2020, Proceedings*, New Orleans, LA, USA, January 20–21, 2020, E. Komendantskaya and Y.A. Liu, eds, Lecture Notes in Computer Science, Vol. 12007, Springer, 2020, pp. 151–167.

[26] C.P. Gomes, A. Sabharwal and B. Selman, Model counting, in: *Handbook of Satisfiability*, A. Biere, M. Heule, H. van Maaren and T. Walsh, eds, Frontiers in Artificial Intelligence and Applications, Vol. 336, 2nd edn, IOS Press, 2021, pp. 993–1014.

[27] A. Haken, The intractability of resolution, *Theor. Comput. Sci.* **39** (1985), 297–308. doi:10.1016/0304-3975(85)90144-6.

[28] M. Heule, W.A. Hunt Jr. and N. Wetzler, Verifying refutations with extended resolution, in: *Automated Deduction – CADE-24–24th International Conference on Automated Deduction*, 2013, pp. 345–359. doi:10.1007/978-3-642-38574-2_24.

[29] M.J.H. Heule, M. Seidl and A. Biere, Solution validation and extraction for QBF preprocessing, *J. Autom. Reason.* **58**(1) (2017), 97–125. doi:10.1007/s10817-016-9390-4.

[30] J. Lagniez and P. Marquis, An improved decision-DNNF compiler, in: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, Melbourne, Australia, August 19–25, 2017, C. Sierra, ed., ijcai.org, 2017, pp. 667–673.

[31] A.L.D. Latour, B. Babaki, A. Dries, A. Kimmig, G.V. den Broeck and S. Nijssen, Combining stochastic constraint optimization and probabilistic programming – from knowledge compilation to constraint solving, in: *Principles and Practice of Constraint Programming – 23rd International Conference, CP 2017, Proceedings*, Melbourne, VIC, Australia, August 28–September 1, 2017, J.C. Beck, ed., Lecture Notes in Computer Science, Vol. 10416, Springer, 2017, pp. 495–511. doi:10.1007/978-3-319-66158-2_32.

[32] J.P. Marques Silva, I. Lynce and S. Malik, Conflict-driven clause learning SAT solvers, in: *Handbook of Satisfiability*, A. Biere, M. Heule, H. van Maaren and T. Walsh, eds, Frontiers in Artificial Intelligence and Applications, IOS Press, 2021.

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

[34] W. Shi, A. Shih, A. Darwiche and A. Choi, On tractable representations of binary neural networks, in: *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020*, Rhodes, Greece, September 12–18, 2020, D. Calvanese, E. Erdem and M. Thielscher, eds, 2020, pp. 882–892.

[35] M. Thurley, sharpSAT – counting models with advanced component caching and implicit BCP, in: *Theory and Applications of Satisfiability Testing – SAT 2006, 9th International Conference, Proceedings*, Seattle, WA, USA, August 12–15, 2006, A. Biere and C.P. Gomes, eds, Lecture Notes in Computer Science, Vol. 4121, Springer, 2006, pp. 424–429. doi:10.1007/11814948_38.

[36] S. Toda, PP is as hard as the polynomial-time hierarchy, *SIAM J. Comput.* **20**(5) (1991), 865–877. doi:10.1137/0220053.

[37] M.Y. Vardi, Boolean satisfiability: Theory and engineering, *Commun. ACM* **57**(3) (2014), 5. doi:10.1145/2578043.

[38] M. Vinyals, Hard examples for common variable decision heuristics, in: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[39] N. Wetzler, M. Heule and W.A. Hunt Jr., DRAT-trim: Efficient checking and trimming using expressive clausal proofs, in: *Theory and Applications of Satisfiability Testing (SAT)*, C. Sinz and U. Egly, eds, Lecture Notes in Computer Science, Vol. 8561, Springer, 2014, pp. 422–429.

[40] E. Zhai, A. Chen, R. Piskac, M. Balakrishnan, B. Tian, B. Song and H. Zhang, Check before you change: Preventing correlated failures in service updates, in: *17th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2020*, Santa Clara, CA, USA, February 25–27, 2020, R. Bhagwan and G. Porter, eds, USENIX Association, 2020, pp. 575–589.