COMPLEXITY OF PROPOSITIONAL PROOFS
UNDER A PROMISE

NACHUM DERSHOWITZ AND IDDO TZAMERET

ABSTRACT. We study—within the framework of propositional proof complexity—the problem of certifying unsatisfiability of CNF formulas under the promise that any satisfiable formula has many satisfying assignments, where "many" stands for an explicitly specified function Λ in the number of variables n. To this end, we develop propositional proof systems under different measures of promises (that is, different Λ) as extensions of resolution. This is done by augmenting resolution with axioms that, roughly, can eliminate sets of truth assignments defined by Boolean circuits. We then investigate the complexity of such systems, obtaining an exponential separation in the average-case between resolution under different size promises:

(1) Resolution has polynomial-size refutations for all unsatisfiable 3CNF formulas when the promise is $\varepsilon \cdot 2^n$, for any constant $0 < \varepsilon < 1$.
(2) There are no sub-exponential size resolution refutations for random 3CNF formulas, when the promise is $2^{\delta n}$, for any constant $0 < \delta < 1$ (and the number of clauses is $O(n^{3/2-\epsilon})$, for $0 < \epsilon < \frac{1}{2}$).

"Goods Satisfactory or Money Refunded"
—The Eaton Promise

1. INTRODUCTION

Demonstrating unsatisfiability of propositional formulas is a fundamental problem in both logic and complexity theory, as well as in hardware and software validation. Any standard sound and complete propositional proof system has the ability to separate the set of unsatisfiable formulas in conjunctive normal form (CNF) from the set of CNF formulas having at least one satisfying assignment, in the sense that every unsatisfiable CNF has a refutation in the system, while no satisfiable CNF has one. Our goal is to develop and study, within the framework of propositional proof complexity, systems that are "sound and complete" in a relaxed sense: they can separate the set of unsatisfiable CNF formulas from the set of CNF formulas having *sufficiently many* satisfying assignments (where the term "sufficiently many" stands for an explicitly given function of the number of variables in the CNF). We call such proof systems *promise refutation systems*, as they are complete and sound for the set of CNF formulas promised to be either unsatisfiable or to have many satisfying assignments.

As the proof systems we develop here are intended for proving *unsatisfiability* of CNF formulas (in other words, to *refute* them, which is the same as validating their negation), throughout this paper we work solely with *refutation* systems, and speak about "refutations"

and "proofs" interchangeably, always intending refutations, unless otherwise stated. In particular, we work with refutation systems that extend the widely studied resolution refutation system.

Our first task is to introduce a natural model for promise propositional refutation systems. This is accomplished by augmenting standard resolution (or any other propositional proof system extending resolution) with an additional collection of axioms, the *promise axioms*. Each refutation in a promise refutation system can make use of at most one promise axiom. The promise axioms are meant to capture the idea that we can ignore or "discard" a certain number of truth assignments from the space of all truth assignments, and still be able to certify (due to the promise) whether or not the given CNF is unsatisfiable. The number of assignments that a promise axiom is allowed to discard depends on the promise we are given, and, specifically, it needs to be less than the number of assignments promised to satisfy a given CNF (unless it is unsatisfiable).

Assuming we have a promise that a satisfiable CNF has more than Λ satisfying assignments, we can discard up to Λ assignments. We refer to Λ as the *promise*. This way, the refutation system is guaranteed not to contain refutations of CNF formulas having more than Λ satisfying assignments, as even after discarding (at most Λ) assignments, we still have at least one satisfying assignment left. On the other hand, any unsatisfiable CNF formula has a refutation in the system, as resolution already has a refutation of it.

Our second task, besides introducing the model of promise refutation systems, is to investigate the basic properties of this model and in particular to determine its average-case proof complexity with respect to different size of promises.

1.1. **Background and Motivation.** In propositional proof complexity theory, it is standard to consider an *abstract* or *formal* propositional proof system (usually called *a Cook-Reckhow proof system*, following [CR79]) as a polynomial-time algorithm A that receives a Boolean formula F (usually in CNF) and a string π over some finite alphabet ("the (proposed) refutation" of F), such that there exists a π with $A(F, \pi) = 1$ if and only if F is unsatisfiable. (A string π for which $A(F, \pi) = 1$ is also called a *witness* for the unsatisfiability of F.) Equipped with this abstract definition of propositional proof systems, showing that *for every* abstract proof system there exists some family of formulas F for which there is no polynomially-bounded family of proofs π of F is equivalent to showing $\mathbf{NP} \neq \mathbf{co\text{-}NP}$.

For this reason (among others), it is customary in proof complexity theory to concentrate on specific (sometimes provably weaker) proof systems for which proofs have a simple structure. This makes the complexity analysis of such proof systems simpler. Prominent examples of such systems are Frege systems and weaker subsystems of Frege, the most notable being the resolution refutation system [Rob65], which also plays an important rôle in many automated theorem provers. In accordance with this, we shall be interested in the present paper not with abstract proof systems (that is, not with finding general witnesses for unsatisfiability, possibly under a promise), but rather with specific and more structured proof systems, and specifically with refutation systems built up as extensions of resolution.

A natural relaxation of the problem of unsatisfiability certification is to require that, if a CNF is satisfiable, then it actually has many satisfying assignments. As mentioned above, we call the specific number of assignments (as a function of the number of variables n) required to satisfy a satisfiable CNF formula, the "promise". Accordingly, one can define an *abstract promise proof system* in an analogous manner to the definition of an abstract proof system.

It is thus natural to ask whether giving such a promise can help in obtaining shorter proofs of unsatisfiability.

In the case of a *big* promise, that is, a constant fraction of the space of all truth assignments ($\Lambda = \varepsilon \cdot 2^n$, for a constant $0 < \varepsilon < 1$), there is already a *deterministic polynomial-time algorithm* for any fixed natural number k that certifies the unsatisfiability of all unsatisfiable kCNF formulas under the promise: the algorithm receives a kCNF that is either unsatisfiable or has more than Λ satisfying assignments and answers whether the formula is unsatisfiable (in case the formula is satisfiable the algorithm provides a satisfying assignment). See [Hir98, Tre04] for such efficient algorithms.[1] This trivially implies the existence of polynomial-size witnesses for any unsatisfiable kCNF under the promise $\varepsilon \cdot 2^n$. But does resolution already admit such short witnesses of unsatisfiability (that is, resolution refutations) under a big promise?

In the case of a *smaller* promise, by which we mean $\Lambda = 2^{\delta n}$ for a constant $0 < \delta < 1$, it is possible to efficiently transform any CNF over n variables to a new CNF with $n' = \lceil n/(1-\delta) \rceil$ variables, such that the original CNF is satisfiable if and only if the new CNF has at least $2^{\delta n'}$ satisfying assignments.[2] Thus, the *worst-case* complexity of certifying CNF unsatisfiability under such a promise is polynomially equivalent to the worst-case complexity of certifying CNF unsatisfiability without a promise. However, it is still possible that a promise of $2^{\delta n}$ might give some advantage (that is, a super-polynomial speedup over refutations without a promise) in certifying the unsatisfiability of certain (but not all) CNF formulas; for instance, in the *average-case*.[3]

Feige, Kim, and Ofek [FKO06] have shown that when the number of clauses is $\Omega(n^{7/5})$ there exist polynomial-size witnesses to the unsatisfiability of 3CNF formulas in the *average-case*. On the other hand, Beame, Karp, Pitassi, and Saks [BKPS02] and Ben-Sasson and Wigderson [BSW01] showed that resolution does not provide sub-exponential refutations for 3CNF formulas in the average-case when the number of clauses is at most $n^{(3/2)-\epsilon}$, for any constant $0 < \epsilon < 1/2$.[4] This shows that general witnessing of 3CNF unsatisfiability is strictly stronger than resolution refutations. But is it possible that, under a promise of $2^{\delta n}$, resolution can do better in the average-case?

There are two main motivations for studying propositional proofs under a given promise and their complexity. The first is to answer the natural question whether CNF unsatisfiability certification enjoys any advantage given a certain promise. As already mentioned, the answer is positive when the promise is a constant fraction of all the truth assignments, and our results

---

[1]In the case the promise is $\Lambda = 2^n/poly(n)$, the algorithm in [Hir98] also gives a deterministic sub-exponential time procedure for unsatisfiability certification of kCNF formulas (for a constant k).

[2] This can be achieved simply by adding new $(n' - n)$ "dummy variables". For instance, by adding the clauses of a tautological CNF in these dummy variables to the original CNF. This way, if the original CNF has at least one satisfying assignment then the new CNF has at least $2^{n'-n} \geq 2^{\delta n'}$ satisfying assignments.

[3]Note that if we add dummy variables to a 3CNF then we obtain an "atypical instance" of a 3CNF. Thus, assuming we have polynomial-size witnesses of unsatisfiability of 3CNF formulas under a small promise in the average-case (that is, the "typical case"), the reduction alone (that is, adding dummy variables) does *not* automatically yield polynomial-size witnesses for 3CNF formulas in the average-case without a promise as well.

[4]Beame *et al.* [BKPS02] showed such a lower bound for $n^{(5/4)-\epsilon}$ number of clauses (for any constant $0 < \epsilon < 1/4$). Ben-Sasson and Wigderson [BSW01] introduced the size-width tradeoff that enables one to prove an exponential lower bound for random 3CNF formulas with $n^{(3/2)-\epsilon}$ number of clauses (for any constant $0 < \epsilon < 1/2$), but the actual proof for this specific clause-number appears in [BS01].

imply that this phenomenon already occurs for resolution. For a small promise of $2^{\delta n}$, we can show that, at least in the case of resolution refutations of most 3CNF formulas (of certain clause-to-variable density), the answer is negative. In fact, we can show that the answer stays negative even when the promise is bigger than $2^{\delta n}$, and specifically when $\Lambda = 2^n/2^{n^\xi}$ for some constant $0 < \xi < 1$. Overall, our results establish the first unsatisfiability certification model in which a promise of a certain given size is known to help (that is, allow more efficient certifications) in the average-case, while promises of smaller sizes do not help.

The second motivation is more intrinsic to proof complexity theory. It is a general goal to develop natural frameworks for propositional proofs that are not sound in the strict sense, but rather possess an approximate notion of soundness (like showing that certain "approximations" give speed-ups). For this purpose, the proof systems we propose formalize — in a natural way — the notion of separating unsatisfiable CNF formulas from those that have many satisfying assignments. Also, the promise axioms we present allow for a natural way of controlling the size of the promise.

1.2. **Overview of the Model of Refutations under a Promise.** We now explain (somewhat informally) what it means to "discard" assignments and how promise axioms formulate the notion of discarding the *correct number* of truth assignments. Essentially, we say that a truth assignment $\bar{a}$ is *discarded* by some Boolean formula if $\bar{a}$ falsifies the formula. More formally, let $X := \{x_1, ..., x_n\}$ be the set of underlying variables of a given CNF, called the *original variables*. Let A be some CNF formula in the variables X, and assume that A also contains variables not from X, called *extension variables*. Let $\bar{a} \in \{0,1\}^n$ be a truth assignment for the X variables, and assume that there is no extension of $\bar{a}$ (assigning values to the extension variables) that satisfies A. Thus, any assignment satisfying A must also satisfy $X \not\equiv \bar{a}$ (that is, $A \models X \not\equiv \bar{a}$), and so any (implicationally) complete proof system can prove $X \not\equiv \bar{a}$ from A, or, in the case of a refutation system, can refute $X \equiv \bar{a}$, given A. In this case, we say that the assignment $\bar{a}$ is *discarded by A*.

The promise axioms we present enjoy two main properties:

(1) They discard assignments from the space of possible assignments to the variables X.
(2) They express the fact that not too many assignments to the variables X are being discarded (in a manner made precise).

The first property is achieved as follows: let C be any Boolean circuit with n output bits. Then we can formulate a CNF formula A (using extension variables) expressing the statement that the output of C is (equal to) the vector of variables X. This enables A to *discard every truth assignment to the variables of X that is outside the image of the Boolean map defined by C*, because, if an assignment $\bar{a}$ to X is not in the image of C, then no extension of $\bar{a}$ can satisfy A—assuming the formulation of A is correct. (For technical reasons, the actual definition is a bit different than what is described here; see Section 3.)

The second property is achieved as follows: assume we can make explicit the statement that the *domain* of the map defined by the Boolean circuit C above is of size at least $2^n - \Lambda$. (See Section 3 for more details.) Then, for the second property to hold, it is sufficient that the axiom formulates the statement that the circuit C defines an *injective* map (and thus the image of the map contains enough truth assignments), which can be done quite naturally.

Given a certain promise and its associated promise axiom, we call a refutation of resolution, augmented with the promise axiom, a *resolution refutation under the (given) promise*.

1.3. **Results.** We show that resolution refutations are already enough to efficiently separate unsatisfiable 3CNF formulas from those 3CNF formulas with an arbitrarily small constant fraction of satisfying assignments. In particular, in Section 4, we show the following:

Main Result 1: Let $0 < \varepsilon < 1$ be a constant and let $\Lambda = \varepsilon \cdot 2^n$ be the given promise. Then *every* unsatisfiable 3CNF with n variables has a polynomial-size (in n) resolution refutation under the promise Λ.

In contrast to the case of a big promise, the results show that, at least for resolution, a small promise of $\Lambda = 2^{\delta n}$ (for any constant $0 < \delta < 1$) does not give any advantage over standard resolution (that is, resolution without the promise axioms) in most cases (that is, in the average-case). Specifically, in Section 5 we show the following:

Main Result 2: Let $0 < \delta < 1$ be a constant and let $\Lambda = 2^{\delta n}$ be the given promise. Then, there is an exponential lower bound on the size of resolution refutations of random 3CNF formulas under the promise Λ, when the number of clauses is $O(n^{3/2-\epsilon})$, for $0 < \epsilon < \frac{1}{2}$.

This lower bound actually applies to a more general model of promise proofs. It remains valid even if we allow (somehow) the promise proofs to discard *arbitrarily chosen* sets of truth assignments (of size $\Lambda = 2^{\delta n}$), and not necessarily those sets that are definable by (small) Boolean circuits. In other words, the lower bound is *semantic*. Furthermore, the lower bound applies even to a bigger promise of $\Lambda = 2^{n-n^\xi}$, for some constant $0 < \xi < 1$.

It is important to note that these two main results show that the decision to discard sets of truth assignments defined by *Boolean circuits* does not affect the results in any way, and thus should not be regarded as a restriction of the model of promise refutations (at least not for resolution). To see this, note that we could allow a promise refutation to discard *arbitrarily chosen* sets of truth assignments (of the appropriate size determined by the given promise), that is, sets of truth assignments that are not necessarily definable by (small) Boolean circuits. However, although this modification strengthens the model, it is not really necessary for the *upper bound* in Main Result 1, as this upper bound is already valid when one discards sets of truth assignments by (small) Boolean circuits. On the other hand, as mentioned above, the *lower bound* in Main Result 2 is already valid when one allows a promise refutation to discard any *arbitrarily chosen* set of truth assignments (of the appropriate size).

2. Preliminaries

2.1. **Notations.** For natural number m, we use $[m]$ to denote the set $\{1, \ldots, m\}$ of naturals.

Let A, B be two propositional formulas. We write $A \equiv B$ as an abbreviation for $(A \to B) \land (B \to A)$. The notation $A \not\equiv B$ abbreviates $\neg(A \equiv B)$. We say that A *semantically implies* B, denoted by $A \models B$, iff every satisfying assignment to A also satisfies B.

A CNF formula over the variables $x_1, \ldots, x_n$ is defined as follows: a *literal* is a variable x_i or its negation $\neg x_i$. A *clause* is a disjunction of literals. We treat a clause as a set of literals, that is, we delete multiple occurrences of the same literal in a clause. A CNF formula is a conjunction of clauses (sometimes treated also as a *set* of clauses, where the conjunction between these clauses is implicit). A kCNF formula is a CNF with all clauses containing k literals each.

The *width* of a clause D is the number of literals in it, denoted $|D|$. The *size* of a CNF formula K is the total number of clauses in it, denoted $|K|$. The *width of a CNF formula* K is the maximum width of a clause in K.

We denote by $K' \subseteq K$ that K' is a sub-collection of clauses from K.

2.2. **Resolution Refutation Systems.** Resolution is a complete and sound proof system for unsatisfiable CNF formulas.

Let C and D be two clauses containing neither x_i nor $\neg x_i$. The *resolution rule* allows one to derive $C \vee D$ from $C \vee x_i$ and $D \vee \neg x_i$. The clause $C \vee D$ is called the *resolvent* of the clauses $C \vee x_i$ and $D \vee \neg x_i$ on the variable x_i, and we also say that $C \vee x_i$ and $D \vee \neg x_i$ were *resolved over* x_i.

The *weakening rule* allows one to derive the clause $C \vee D$ from the clause C, for any two clauses C, D.

Definition 2.1 (Resolution). A *resolution proof of the clause D from a CNF formula K* is a sequence of clauses $D_1, D_2, \ldots, D_\ell$, such that: (1) each clause D_j is either a clause of K or a resolvent of two previous clauses in the sequence or derived by the weakening rule from a previous clause in the sequence; (2) the last clause $D_\ell = D$. The size of a resolution proof is the total number of clauses in it. The width of a resolution proof is the maximal width of a clause in it. A *resolution refutation* of a CNF formula K is a resolution proof of the empty clause $\square$ from K. (The empty clause stands for FALSE; that is, the empty clause has no satisfying assignments.)

Let K be an unsatisfiable CNF formula. The *resolution refutation size of K* is the minimal size of a resolution refutation of K and is denoted $S(K)$. Similarly, the *resolution refutation width of K* is the minimal width of a resolution refutation of K and is denoted $w(K)$. If K has a polynomial-size resolution refutation we say that resolution can *efficiently certify* the unsatisfiability of K. Similarly, if the clause D has a polynomial-size resolution proof from K we say that D *is efficiently provable from K*.

2.3. **Size-Width Tradeoffs.** We recall now the approach for proving size lower bounds on resolution refutations developed by Ben-Sasson and Wigderson [BSW01]. The basic idea is that a lower bound on the resolution refutation width of a CNF formula K implies a lower bound on the resolution refutation size of K:

Theorem 1 ([BSW01]). *Let K be a CNF formula of width r, then*

$$S(K) = \exp\left(\Omega\left(\frac{(w(K) - r)^2}{n}\right)\right).$$

2.4. **Boolean Circuit Encoding.** The promise axioms we introduce use Boolean circuits to define the sets of assignments to be discarded (see Section 3). Therefore, as resolution operates only with clauses, we need to encode Boolean circuits as collections of clauses (CNF formulas). We assume that all Boolean circuits use only three gates: $\vee, \wedge, \neg$ (though this is not necessary) where $\vee$ (denoting OR) and $\wedge$ (denoting AND) have fan-in 2 and $\neg$ (denoting NOT) has fan-in 1. Let $C : \{0,1\}^m \to \{0,1\}^n$ be a Boolean circuit with m input bits and n output bits. Let $\overline{W} = \{w_1, \ldots, w_m\}$ be the m input variables of C and let X denote the n variables $\{x_1, \ldots, x_n\}$. We consider the n output bits of C as the outputs of n distinct circuits $C_1(\overline{W}), \ldots, C_n(\overline{W})$ in the $\overline{W}$ variables, and we write $C(\overline{W}) \equiv X$ to mean that X

equals the output of $C(\overline{W})$ (that is, $C_1(\overline{W}) \equiv x_1 \wedge \cdots \wedge C_n(\overline{W}) \equiv x_n$). This notation can be extended in a similar manner to $C(\overline{W_1}) \equiv C'(\overline{W_2})$ and $C(\overline{W_1}) \not\equiv C'(\overline{W_2})$.

There exists a CNF formula F (in both the $\overline{W}$ variables and new extension variables) that *encodes the circuit* C. This means that there are n new extension variables (among other extension variables) $y_1, \ldots, y_n$ in F, such that for all assignments $\overline{a}$ (where $\overline{a}$ is an assignment to *all* the variables in F, including all extension variables): if $F(\overline{a}) = 1$ then $C(w_1(\overline{a}), \ldots, w_m(\overline{a})) = (y_1(\overline{a}), \ldots, y_n(\overline{a}))$, where we denote by $w_i(\overline{a})$ the truth value of w_i under the assignment $\overline{a}$. In other words, F expresses the fact that $y_1, \ldots, y_n$ are the output bits of C. Moreover, if C is of size polynomial in n then F is also of size polynomial in n. We denote by $\|C(\overline{W})\|$ the CNF formula F that encodes $C(\overline{W})$.

For most purposes, we will not need an explicit description of how the encoding of Boolean circuits as CNF formulas is done through $\|C(\overline{W})\|$. Nevertheless, in Section 4 we need to ensure that resolution can efficiently prove several basic facts about the encoded circuits. For this reason, and for the sake of concreteness of the promise axioms (Definitions 3.3 and 3.4) we provide the precise definition of the encoding in the Appendix (Section A.1), in addition to proving some of the encoding's basic (proof theoretical) properties.

3. Promise Proof Systems

In this section we define precisely the model of refutations under a promise. We work with the resolution refutation system as our underlying system and augment it with a new set of axioms that we call the *promise axioms*. We call this proof system *promise resolution*. The promise axioms are meant to express the fact that we can discard a certain number of truth assignments from the space of all truth assignments and still be able to certify (due to the promise) whether the input CNF is unsatisfiable or not. Each promise resolution refutation can use at most one promise axiom.

From now on, throughout the paper, unless otherwise stated, we shall assume that the underlying variables of the CNF formulas that are meant to be refuted are taken from the set $X := \{x_1, \ldots, x_n\}$. The X variables are called the *original variables*. Any other variable that appears in a (promise resolution) refutation is called an *extension variable*.

Definition 3.1 (CNF formulas under a promise). Let Λ be some function of n (the number of X variables) such that $0 \le \Lambda(n) \le 2^n$. The function Λ is called the *promise*. The set of *CNF formulas under the promise* Λ consists of all CNF formulas in the X variables that are either unsatisfiable or have more then $\Lambda(n)$ satisfying assignments (for $n = |X|$).

The refutation systems we build are sound and complete for the set of CNF formulas under a (given) promise. That is, every unsatisfiable CNF formula has a refutation in the system (this corresponds to completeness), while no CNF having n variables and more than $\Lambda(n)$ satisfying assignments has a refutation in it (this corresponds to soundness under the promise). Soundness (under the promise) is achieved by requiring that resolution should *prove the fact that we discard the right number of assignments.*

Definition 3.2 (Assignment discarding). Let A be a CNF in the X variables that can contain (but does not necessarily contain) extension variables (that is, variables not from X). We say that an assignment to the X variables $\overline{a}$ is *discarded* by A if there is no extension of $\overline{a}$ (to the extension variables in A) that satisfies A.

3.1. Promise Axioms.

3.1.1. *Big promise.* We first concentrate on a promise of a *constant fraction of assignments* (for a smaller promise the axiom is similar; see below).

Let the promise (Definition 3.1) be $\Lambda = \varepsilon \cdot 2^n$, for a constant $0 < \varepsilon < 1$ (we fix this Λ throughout this subsection), and let $r = \lceil \log(1/\varepsilon) \rceil$ and $t = 2^r - 1$. Let C be a sequence of Boolean circuits $C := (C^{(1)}, \ldots, C^{(t)})$. Assume that each $C^{(i)}$ has $n - r$ input bits and n output bits and computes the Boolean map $f_i : \{0,1\}^{n-r} \to \{0,1\}^n$. Assume further that the f_i's are all injective maps and that the images of all these maps are pairwise disjoint. Denote by $\mathrm{Im}(f_i)$ the image of the map f_i. We call the union $\cup_{i=1}^t \mathrm{Im}(f_i)$ *the image of C* and denote it by $\mathrm{Im}(C)$. By definition $2^{n-r} = 2^{n-\lceil \log(1/\varepsilon) \rceil}$, and since

$$\varepsilon \cdot 2^{n-1} = 2^{n-1-\log(1/\varepsilon)} \le 2^{n-\lceil \log(1/\varepsilon) \rceil} \le 2^{n-\log(1/\varepsilon)} = \varepsilon \cdot 2^n \,,$$

we have

$$\frac{\varepsilon}{2} \cdot 2^n \le 2^{n-r} \le \varepsilon \cdot 2^n \,.$$

By the injectivity and pairwise disjointness of the images of the f_i's we have $|\mathrm{Im}(C)| = t \cdot 2^{n-r} = (2^r - 1) \cdot 2^{n-r} = 2^n - 2^{n-r}$, and so

$$2^n - \varepsilon \cdot 2^n \le |\mathrm{Im}(C)| \le 2^n - \frac{\varepsilon}{2} \cdot 2^n \,. \tag{1}$$

Therefore, *we can treat $\mathrm{Im}(C)$ as the set of all possible truth assignments for the original variables X, without losing soundness*: if K is unsatisfiable then there is no assignment in $\mathrm{Im}(C)$ that satisfies K; and if K is satisfiable then according to the promise it has more than Λ satisfying assignments, which means that there is at least one assignment in $\mathrm{Im}(C)$ that satisfies K. This idea is formulated as a propositional formula as follows:

Definition 3.3 (Promise Axiom for $\Lambda = \varepsilon \cdot 2^n$)**.** Let the promise be $\Lambda = \varepsilon \cdot 2^n$, for a constant $0 < \varepsilon < 1$, and let $r = \lceil \log(1/\varepsilon) \rceil$ and $t = 2^r - 1$. Let C be a sequence of Boolean circuits $C := (C^{(1)}, \ldots, C^{(t)})$. Assume that each $C^{(i)}$ has $n-r$ input bits and n output bits and let $\overline{W_1}$ and $\overline{W_2}$ be two disjoint sets of $n-r$ extension variables each. The promise axiom $PRM_{C,\Lambda}$ is the CNF encoding of the following Boolean formula (the encoding of this formula as a CNF is shown explicitly in Section A.1, and uses encoding of Boolean *circuits* as illustrated in Section 2.4; the latter is defined explicitly in Definition A.1):

$$\left(\bigwedge_{i=1}^t \left(C^{(i)}(\overline{W_1}) \equiv C^{(i)}(\overline{W_2}) \to \overline{W_1} \equiv \overline{W_2} \right) \wedge \bigwedge_{1 \le i < j \le t} C^{(i)}(\overline{W_1}) \not\equiv C^{(j)}(\overline{W_2}) \right)$$

$$\longrightarrow \bigvee_{i=1}^t C^{(i)}(\overline{W_1}) \equiv X.$$

The promise axiom $PRM_{C,\Lambda}$ expresses the fact that if each circuit in C computes an injective map (this is formulated as $\wedge_{i=1}^t (C^{(i)}(\overline{W_1}) \equiv C^{(i)}(\overline{W_2}) \to \overline{W_1} \equiv \overline{W_2})$), and if the images of the maps computed by each pair of circuits in C are disjoint (this is formulated as $\wedge_{1 \le i < j \le t} C^{(i)}(\overline{W_1}) \not\equiv C^{(j)}(\overline{W_2})$), then we can assume that the assignments to the original variables X are taken from the image of C (this is formulated as $\vee_{i=1}^t C^{(i)}(\overline{W_1}) \equiv X$). The fact that the image of C is of size at least $2^n - \Lambda$ is expressed (due to Equation (1)) by the number of input bits (that is, $n - r$) of each circuit in C and the number of circuits in C

8

(that is, t). Also note that the promise axiom is of polynomial-size as long as the circuits in C are (since $1/\varepsilon$ is a constant).

The following claim shows that the promise axioms are sound with respect to the promise Λ, in the sense that they do not discard too many truth assignments:

Proposition 2. *The promise axiom* $\mathrm{PRM}_{C,\Lambda}$ *discards at most Λ truth assignments. That is, there are at most Λ distinct assignments $\overline{a}$ to the X variables such that $\mathrm{PRM}_{C,\Lambda} \models X \not\equiv \overline{a}$.*

Proof. Assume that some Boolean map computed by some circuit in C is not injective. Then any assignment to the X variables has an extension ρ (to the extension variables in the promise axiom) that falsifies the premise of the main implication in $\mathrm{PRM}_{C,\Lambda}$ and thus ρ satisfies $\mathrm{PRM}_{C,\Lambda}$. Therefore no assignments to X are discarded.

Similarly, assume that the images of some pair of maps computed by two circuits in C are not disjoint. Then, again, any assignment to the X variables has an extension that satisfies $\mathrm{PRM}_{C,\Lambda}$, and so no assignments to X are discarded.

Assume that all the Boolean maps computed by circuits in C *are* injective and have pairwise disjoint images. Then every assignment satisfies the premise of the main implication in the promise axiom $\mathrm{PRM}_{C,\Lambda}$. Therefore, it suffices to show that the consequence of the main implication of the axiom (that is, $\vee_{i=1}^{t} C^{(i)}(\overline{W_1}) \equiv X$) discards at most Λ assignments to the X variables. By definition (of the encoding of the circuits) for all assignments $\overline{a}$ to the X variables that are in $\mathrm{Im}(C)$ there is an extension of $\overline{a}$ that satisfies $\vee_{i=1}^{t} C^{(i)}(\overline{W_1}) \equiv X$. Now, all the circuits $C^{(i)}$ compute injective maps with pairwise disjoint images, and thus by Equation (1) there are at least $2^n - \Lambda$ distinct elements (that is, assignments) in $\mathrm{Im}(C)$. Hence, at least $2^n - \Lambda$ assignments to the X variables are not discarded. $\qquad\square$

3.1.2. *Smaller promise.* Here we formulate promise axioms for promises smaller than $\varepsilon \cdot 2^n$. Specifically, we work with a promise of $\Lambda = 2^{\delta n}$ for a constant $0 < \delta < 1$. For such a promise, the promise axiom is similar to Definition 3.3, except that the number of input bits of each circuit in C needs to be modified accordingly. (We use the same terminology as that used above for the Big Promise.)

Definition 3.4 (Promise Axiom for $\Lambda = 2^{\delta n}$). Let the promise be $\Lambda = 2^{\delta n}$, for a constant $1 < \delta < 1$, and let $t = \lceil (1 - \delta)n \rceil$. Let C be a sequence of Boolean circuits $C := (C^{(1)}, \ldots, C^{(t)})$. Assume that for each $1 \leq i \leq t$ the circuit $C^{(i)}$ has $n - i$ input bits and n output bits. Let $\overline{W_1}, \ldots, \overline{W_t}$ and $\overline{W_1'}, \ldots, \overline{W_t'}$ be $2t$ disjoint sets of extension variables[5], where for all $1 \leq i \leq t$, W_i, W_i' consist of $n - i$ variables each. The promise axiom $PRM_{C,\Lambda}$ is a CNF encoding of the following Boolean formula:

$$\left(\bigwedge_{i=1}^{t} \left(C^{(i)}(\overline{W_i}) \equiv C^{(i)}(\overline{W_i'}) \rightarrow \overline{W_i} \equiv \overline{W_i'} \right) \wedge \bigwedge_{1 \leq i < j \leq t} C^{(i)}(\overline{W_i}) \not\equiv C^{(j)}(\overline{W_j}) \right)$$
$$\longrightarrow \bigvee_{i=1}^{t} C^{(i)}(\overline{W_i}) \equiv X.$$

Note that the promise axiom is of polynomial size as long as the circuits in C are (since $t \leq n$).

Also note that the proof of Proposition 2 did not use the parameters r and t (which determine the number of input bits in the circuits in C and the number of circuits in C,

---

[5]We have not been very economical in adding extension variables here; but this is not essential.

respectively) but only the size $|\text{Im}(C)|$. Thus, the same claim holds also for the promise axiom in Definition 3.4, which means that this promise axiom discards at most $2^n - |\text{Im}(C)|$ truth assignments, for some sequence of circuits in C that compute injective maps with pairwise disjoint images. Therefore, we need to verify that $|\text{Im}(C)| \geq 2^n - \Lambda$, for all C that consists of circuits computing injective maps with pairwise disjoint images.

Notice that for all $1 \leq i \leq t$ the circuit $C^{(i)}$ computes a Boolean map, denoted f_i, such that $f_i : \{0,1\}^{n-i} \rightarrow \{0,1\}^n$. Assume that all the f_i's are injective and that the images of each pair of functions f_i, f_j, for $1 \leq i \neq j \leq t$, are disjoint. Then, we have:

$$\begin{aligned} |\text{Im}(C)| &= \left(\frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \cdots + \frac{1}{2^t} \right) \cdot 2^n = \left(1 - \frac{1}{2^t} \right) \cdot 2^n \qquad (2) \\ &= 2^n - 2^{n - \lceil (1-\delta)n \rceil} \geq 2^n - 2^{\delta n} = 2^n - \Lambda \end{aligned}$$

Also note that $|\text{Im}(C)| \leq 2^n - 2^{\delta n - 1}$ and so if the circuit in C are injective with pairwise disjoint images then $\text{PRM}_{C,\Lambda}$ discards *at least* $2^{\delta n}/2$ truth assignments.

3.2. Promise Resolution.

Definition 3.5 (Promise resolution). Let Λ be the promise (Definition 3.1) and let K be a CNF in the X variables. A *promise resolution (under the promise Λ) proof of the clause D from a CNF formula K* is a sequence of clauses $D_1, D_2, \ldots, D_\ell$ such that:

(1) Each clause D_j is either a clause of K or a clause of a promise axiom $PRM_{C,\Lambda}$ (where $PRM_{C,\Lambda}$ is either a big or a smaller promise axiom as defined in Definitions 3.3 and 3.4 and C is an arbitrary sequence of circuits with the prescribed input and output number of bits) or a resolvent of two previous clauses in the sequence;
(2) The sequence contains (the clauses of) at most one promise axiom;
(3) The last clause $D_\ell = D$.

The *size*, *width* and *refutations* of promise resolution is defined the same as in resolution.

Note that promise resolution is a Cook-Reckhow proof system (see the first paragraph in Section 1.1 for a definition): it is possible to efficiently verify whether a given CNF is an instance of the promise axiom, and hence to verify whether a sequence of clauses constitute a legitimate promise refutation. This can be done by "decoding" the CNF that encodes the promise axiom $\text{PRM}_{C,\Lambda}$ and then checking that each circuit in C has the right number of input and output bits (we discuss this issue in some more detail in the appendix).

Proposition 3. *Let Λ be the promise (where Λ is either $\varepsilon \cdot 2^n$ or $2^{\delta n}$, for $0 < \varepsilon, \delta < 1$). Then promise resolution under the promise Λ is a sound and complete proof system for the set of CNF formulas under the promise Λ. In other words, every unsatisfiable CNF has a promise resolution refutation and every CNF that has more than Λ satisfying assignments does not have promise resolution refutations.*

Proof. Completeness stems from completeness of resolution. Soundness under the promise Λ stems from Proposition 2 (which, by the notes after Definition 3.4, holds for both the big and the smaller promise axioms). $\square$

3.3. Discussion. Let K be an unsatisfiable CNF formula in n variables, $\mathrm{PRM}_{C,\Lambda}$ a promise axiom (where the circuits in C all compute injective and pairwise disjoint Boolean maps) and let $\mathrm{Im}(C) \subseteq \{0,1\}^n$ be such that $|\mathrm{Im}(C)| \geq 2^n - \Lambda$. Then, intuitively, one can think of a promise resolution refutation of K that makes use of the axiom $\mathrm{PRM}_{C,\Lambda}$ as containing two parts (in what follows we say that resolution proves a certain statement or formula, meaning that resolution *refutes the negation of the formulas*):

(i) A resolution proof that $|\mathrm{Im}(C)| \geq 2^n - \Lambda$. This corresponds to proving in resolution the (CNF encoding of the) premise of the main implication in the promise axioms.

(ii) A resolution "refutation" of K where the space of truth assignments is restricted to $\mathrm{Im}(C)$. This corresponds to the refutation of the CNF consisting of the clauses of both K and the clauses of the (CNF encoding of the) conclusion of the main implication in the promise axioms: $\vee_{i=1}^{t} C^{(i)}(\overline{W}) \equiv X$.

Note that if we want to consider promise resolution as having only part (i), then we can modify (actually, strengthen) the promise axiom to consists of only $\vee_{i=1}^{t} C^{(i)}(\overline{W}) \equiv X$. However, this choice means losing the soundness of the proof system under the promise (that is, the soundness with respect to CNF formulas under a promise as defined in Definition 3.1), since we do not have any guarantee that the circuit C discards at most Λ assignments (and so CNF formulas with more than Λ satisfying assignments might have refutations in such a system).

It is possible to use any number of axioms of the form $C^{(i)}(\overline{W}) \equiv X$, as long as resolution can prove both the injectivity of each of the maps computed by the circuits $C^{(i)}$ introduced and the pairwise disjointness of these maps (as formulated by a propositional formula similar to the formulation in the promise axioms), and provided that the circuits $C^{(i)}$ have number of input bits that induce the right size of domains (that is, that the total size of their domains is at least $2^n - \Lambda$).

It is also possible to modify the promise axioms to suit any chosen size of promise Λ (possibly, only an approximation of Λ). This can be achieved by choosing a sequence of circuits with the appropriate size of domain (explicitly expressed by the number of input bits in each circuit in the sequence, and the total number of circuits).

Some comments about the formulation of the promise axioms are in order.

Comment 1. Note that we could not use only a single circuit C in the promise axioms (in contrast to a sequence of circuits), because that way we would not have the possibility of controlling the size of the domain of C and efficiently verifying that this size is the correct one inside resolution. To see this, notice that if the number of input variables to C is n (the number of original variables) and the map computed by C is (provably) injective then C does not discard any assignment. If, on the other hand, the number of input variables to C is less than n, then C discards at least half the truth assignments, which might be too many.

Comment 2. Also note that in order to discard assignments we cannot use a seemingly more natural axiom of the form $C(\overline{W}) \not\equiv X$ for some circuit C (with domain of size Λ). The reason is that this would not discard assignments in the image of C: it is not necessarily true that $C(\overline{W}) \not\equiv X \models X \not\equiv \overline{b}$ for all $\overline{b} \in \{0,1\}^n$ such that $b \in \mathrm{Im}(C)$ (notice that even for such a $\overline{b}$ there might be some assignment $\overline{a}$ for which $C(\overline{a}) \not\equiv \overline{b}$).

On the other hand, Jan Krajíček [Kra07] observed that it is possible to discard assignments by an axiom of the form $C(\overline{W}) \not\equiv X$, where C is a fixed circuit with domain of size at most

11

Λ (that is, it has $k < n$ number of input bits, where $2^k \leq \Lambda$), and where the rule of using this axiom is that we can introduce any instance of $C(\overline{W}) \not\equiv X$ where *all the variables in $\overline{W}$ are substituted by constants* $0, 1$ *and variables from* X. This choice of axioms simplifies somewhat the actual formulation of the promise axioms, as it does not require that C computes an injective Boolean map. However, a possible drawback of such a formulation is the following: it is possible that for certain circuits (of the appropriate number of input and output bits) we shall need to use exponentially many such axiom instances to discard *all* (or most of) the assignments pertaining to the image of the circuits. In contrast to this, our formulation of the promise axioms above enables a single instance of a promise axiom using *any* circuit (more correctly, a sequence of circuits of the appropriate number of input and output bits) to discard *all* the assignments outside the image of the circuit.

4. Big Promise: Upper Bound

In this section, we show that under the promise $\Lambda = \varepsilon \cdot 2^n$, for any constant $0 < \varepsilon < 1$, resolution can efficiently certify the unsatisfiability of all unsatisfiable 3CNF formulas. The proof method resembles the algorithm presented by Trevisan [Tre04]. For a constant k, this algorithm receives a kCNF formula K and deterministically approximates the fraction of satisfying assignments of K within an additive error of ε. The running time of the algorithm is linear in the size of K and polynomial in $1/\varepsilon$.

The idea behind the refutations is based on the following observation: given an unsatisfiable 3CNF formula K and a constant c, either there are $3(c-1)$ variables that hit[6] all the clauses in K or there are at least c clauses in K over $3c$ *distinct* variables denoted by K' (that is, each variable in K' appears only once). In the first case, we can consider all the possible truth assignments to the $3c$ variables inside resolution: if K is unsatisfiable then any such truth assignment yields an unsatisfiable 2CNF formula, which can be efficiently refuted in resolution (cf. [Coo71]). In the second case, we can make use of a promise axiom to efficiently refute K' (this set of clauses has less then Λ satisfying assignments, for sufficiently large c). Specifically, in the second case, we construct a sequence of small circuits C for which any satisfying assignment for K' is *provably in resolution* (with polynomial-size proofs) outside the image of C.

The following is the main result of this section:

Theorem 4. *Let $0 < \varepsilon < 1$ be a constant and let $\Lambda = \varepsilon \cdot 2^n$ be the given promise. Then every unsatisfiable 3CNF with n variables has a polynomial-size (in n) resolution refutation under the promise Λ.*

The proof of this theorem is given as a sequence of lemmas in the rest of this section.

Lemma 5. *Let K be a 3CNF formula. For every integer c one of the following holds:*
  (i) *There is a set of at most $3(c-1)$ variables that hit all the clauses in K.*
  (ii) *There is a sub-collection of clauses from K, denoted K', with at least c clauses and where each variable appears only once in K'.*

---

[6]A set of variables S that "hit all the clauses in a CNF formula K" is a set of variables for which every clause in K contains some
variable from S.

Proof. Assume that $c > 2$ (other wise the lemma is trivial). Suppose that there is no set of at most $3(c-1)$ variables that hit all the clauses in K and let D_1 be some clause in K. Then, there ought to be a clause D_2 from K that contains 3 variables that are not already in D_1 (or otherwise, the 3 (distinct) variables in D_1 hit all the clauses in K, which contradicts the assumption). In a similar manner we can continue to add new clauses from K until we reach a set of c clauses $D_1, D_2, \ldots, D_c$, where no variable appears more than once in this set of clauses. $\qquad\square$

For the rest of this section, we fix the value of c to be the following constant (independent of n):

$$c = \lceil \log_{7/8}(\varepsilon/2) \rceil . \tag{3}$$

We show that for any unsatisfiable 3CNF K with n variables, in each of the two cases in Lemma 5 (for c as in (3)), there is a polynomial-size refutation (in n) of K under the promise $\varepsilon \cdot 2^n$.

Case (i) in Lemma 5 holds. The following lemma suffices to efficiently refute the 3CNF K in this case:

Lemma 6. *Let K be an unsatisfiable 3CNF formula in the X variables (where $n = |X|$). Assume that there is a set $S \subseteq X$ of at most $3(c-1)$ variables that hit all the clauses in K. Then, there is a polynomial-size (in n) resolution refutation of K.*

Proof sketch: We simply run through all truth assignments to the variables in S (since $|S| \leq 3(c-1)$, there are only constant number of such truth assignments). Under each truth assignment to the S variables, K becomes an unsatisfiable 2CNF. It is known that any unsatisfiable 2CNF has a polynomial-size resolution refutation (cf. [Coo71]). Thus, we can refute K with a polynomial-size resolution refutation. $\qquad\square$

Case (ii) in Lemma 5 holds. The two lemmas that follow suffice to conclude that the 3CNF K has a polynomial-size (in n) resolution refutation under the promise (in this case).

Lemma 7. *Let K' be the 3CNF from case (ii) in Lemma 5, with variables $x_1, \ldots, x_{3c}$. Then,*
  (a) *K' has less than an $\varepsilon/2$ fraction of satisfying assignments; and*
  (b) *K' has a constant-size (independent of n) resolution refutation under the promise $\varepsilon \cdot 2^{3c}$ (that is, there exist a promise axiom A, such that $A \cup K'$ has a constant-size resolution refutation).[7]*

Proof. (a) Observe that the fraction of distinct assignments that satisfy K' is equal to the probability (over all truth assignments to K') that a uniformly chosen random truth assignment satisfies all the c clauses in K', which is equal to

$$\left(\frac{7}{8}\right)^c = \left(\frac{7}{8}\right)^{\lceil \log_{7/8}(\varepsilon/2) \rceil} \leq \left(\frac{7}{8}\right)^{\log_{7/8}(\varepsilon/2)} = \varepsilon/2 . \tag{4}$$

(b) By (a), K' has at most $\frac{\varepsilon}{2} \cdot 2^{3c}$ satisfying assignment. By Equation (1) in Section 3.1.1, there exists a sequence of (injective and with pairwise disjoint images) circuits C such that

---

[7]We abuse the terminology here. We defined a refutation of a CNF under a promise only for unsatisfiable CNFs, though here K' *is satisfiable*. The point here is that there exits an appropriate promise axiom that can discard all the satisfying assignments of K', and so there is a refutation of K' given the axiom.

the promise axiom $\mathrm{PRM}_{C,\varepsilon\cdot2^{3c}}$ discards *all* the $\frac{\varepsilon}{2}\cdot 2^{3c}$ satisfying truth assignments of K'. Thus, (the CNF consisting of the clauses of both) K' (which contains only the variables $x_1,\ldots,x_{3c}$) and $\mathrm{PRM}_{C,\varepsilon\cdot2^{3c}}$ has resolution refutation. Since c is a constant (independent of n), this refutation is also of constant size. $\qquad\square$

We now show that the refutation of K' from Lemma 7 can be extended with only a polynomial (in n) increase in size to a refutation of the same K' under the promise $\varepsilon\cdot 2^n$, that is, when the set of formal variables in the refutation is extended from $\{x_1,\ldots,x_{3c}\}$ to $\{x_1,\ldots,x_n\}$ (for $n>3c$) (there are also extension variables from the promise axioms). This will conclude the proof of Theorem 4.

Lemma 8. *Let K' be the 3CNF from case (ii) in Lemma 5, with variables $x_1,\ldots,x_{3c}$. Then, for any $n\geq 3c$, K' has a resolution refutation of polynomial-size (in n) under the promise $\varepsilon\cdot 2^n$, where the formal variables in the refutation are $x_1,\ldots,x_n$ (as well as the extended variables in the promise axioms).*[8]

Proof. By Lemma 7 Item (b), there exists a promise axiom $\mathrm{PRM}_{C,\varepsilon\cdot2^{3c}}$ consisting of a sequence of injective and with pairwise disjoint images circuits C such that $\mathrm{PRM}_{C,\varepsilon\cdot2^{3c}}$ discards *all* the (at most $\frac{\varepsilon}{2}\cdot 2^{3c}$) satisfying truth assignments of K'. Thus, letting $r=\lceil\log(1/\varepsilon)\rceil$ and $t=2^{r-1}$, we have that $C=(C^{(1)},\ldots,C^{(t)})$ such that $C^{(i)}(w_1,\ldots,w_{3c-r}):\{0,1\}^{3c-r}\to\{0,1\}^{3c}$, for all $i\in[t]$. Let $\overline{W}_0=\{w_1,\ldots,w_{3c-r}\}$. We write C as follows:

$$C_1^{(1)}(\overline{W}_0),\ \ldots,\ C_{3c}^{(1)}(\overline{W}_0)$$
$$\vdots$$
$$C_1^{(t)}(\overline{W}_0),\ \ldots,\ C_{3c}^{(t)}(\overline{W}_0)\,,$$

where $C_j^{(i)}$ denotes the jth output of the ith circuits $C^{(i)}$ in C. Let $B\subseteq\{0,1\}^{3c}$ be the set of truth assignments that falsify K'. By assumption that the circuits in C discard all the satisfying truth assignments in K', $\mathrm{Im}(C)\subseteq B$.

We now construct a sequence of circuits C' for the promise axiom $\mathrm{PRM}_{C',\varepsilon\cdot2^n}$ that discard all the satisfying truth assignments of K' considered as a 3CNF over the variables $x_1,\ldots,x_n$. The sequence C' consists of the following circuits:

$$C_1^{(1)}(\overline{W}_0),\ \ldots,\ C_{3c}^{(1)}(\overline{W}_0),$$
$$C_{3c+1}^{(1)}(w_{3c-r+1})=w_{3c-r+1},\ \ldots,\ C_n^{(1)}(w_{n-r})=w_{n-r}$$
$$\vdots\qquad\qquad\qquad\qquad\vdots\qquad\qquad\qquad\qquad(5)$$
$$C_1^{(t)}(\overline{W}_0),\ \ldots,\ C_{3c}^{(t)}(\overline{W}_0),$$
$$C_{3c+1}^{(t)}(w_{3c-r+1})=w_{3c-r+1},\ \ldots,\ C_n^{(t)}(w_{n-r})=w_{n-r},$$

where $C_j^{(i)}(w_k)=w_k$ denotes the fact that $C_j^{(i)}$ outputs the (input) variable w_k (in which case we assume that the circuit $C_j^{(i)}$ consists of only a single gate: the variable w_k).

Note that the circuits in C' indeed compute t injective Boolean maps (each with domain $\{0,1\}^{n-r}$ and range $\{0,1\}^n$) that have pairwise disjoint images and that C' discards all the satisfying truth assignments of K'. Disjointness of images stems from the fact that the original sequence of circuits C all have disjoint images, and injectivity stems from the fact

---

[8]Again, as in Lemma 7, we abuse the terminology of "a refutation under a promise" here.

that the original sequence of circuits C are all injective and that for each $1 \leq i \leq t$, the Boolean map computed by $(C_{3c+1}^{(i)}, \ldots, C_n^{(i)})$ is exactly the identity map $\mathrm{id} : \{0,1\}^{n-3c} \to \{0,1\}^{n-3c}$, over the variables $w_{3c-r+1}, \ldots, w_{n-r}$.

To complete the proof of Lemma 8, we need to show that there are resolution proofs of polynomial-size in n for the formulas expressing that the circuits in C' all compute injective Boolean maps and have pairwise disjoint images (as well as to efficiently refute K' when assuming that X can take only assignments from the image of C'). This is done in the following claim:

Claim 1. Let C' be the sequence of circuits as devised above, and let $\mathrm{PRM}_{C', \varepsilon \cdot 2^n}$ be the corresponding promise axiom. Then, there is a resolution refutation of $K' \cup \mathrm{PRM}_{C', \varepsilon \cdot 2^n}$ with size polynomial in n.

The proof of this claim follows by considering the encoding of the promise axiom $\mathrm{PRM}_{C', \varepsilon \cdot 2^n}$ via the encoding scheme in the Appendix (Section A.1) and showing how resolution can prove the empty clause from K' and this axiom. Here we shall sketch the proof in a less formal manner; more details can be found in the appendix.

Proof sketch of Claim 1: First we need resolution to prove (formally, to refute its negation; that is, derive the empty clause) from the (encoding of the) premise of the main implication in $\mathrm{PRM}_{C', \varepsilon \cdot 2^n}$. This breaks into two refutations corresponding to the two conjuncts in the premise of the promise axiom $\mathrm{PRM}_{C', \varepsilon \cdot 2^n}$: $\bigwedge_{i=1}^{t} \left(C^{(i)}(\overline{W}_1) \equiv C^{(i)}(\overline{W}_2) \to \overline{W}_1 \equiv \overline{W}_2 \right)$ and $\bigwedge_{1 \leq i < j \leq t} C^{(i)}(\overline{W}_1) \not\equiv C^{(j)}(\overline{W}_2)$.

For the first part, we need to refute the statement expressing that C' contains some circuit $C^{(i)}$ that computes a non-injective Boolean map. This can be efficiently refuted in resolution: assume (inside resolution) that $C^{(i)}(\overline{W}_1) \equiv C^{(i)}(\overline{W}_2)$, for some $1 \leq i \leq t$, then by (5) we can prove (inside resolution; that is, refute its negation) with a polynomial-size in n refutation that for all $d - r + 1 \leq j \leq n - r$ it happens that $w_j^{(1)} \equiv w_j^{(2)}$ (where $w_j^{(1)}$ is the jth variable in $\overline{W}_1$, and $w_j^{(2)}$ is the jth variable in $\overline{W}_2$) (see details in the appendix, and in particular Section A.1.3). Thus, it remains to refute the statement that for some $1 \leq j \leq d - r$ it happens that $w_j^{(1)} \not\equiv w_j^{(2)}$. This is indeed a contradiction by definition of the circuits in the (original) C (as they compute injective maps). Since this statement is formulated by the circuits in C, then by assumption such a contradiction can be refuted with a constant-size (independent of n) resolution refutation.

The disjointness of the images of the (maps computed by the) circuits in C' is also efficiently provable inside resolution in a similar manner, and we omit the details.

Therefore, we arrive (inside resolution) at the consequence of the main implication in the promise axiom: $\bigvee_{i=1}^{t} C^{(i)}(\overline{W}_1) \equiv X$. It remains to refute the conjunction of K' and

$$\bigvee_{i=1}^{t} C^{(i)}(\overline{W}_1) \equiv X. \tag{6}$$

The 3CNF T contains only the variables $x_1, \ldots, x_{3c}$. Consider the part of the (CNF encoding) of (6) that consists of (the encoding of) the circuits from C' that output to the variables $x_1, \ldots, x_{3c}$. This encoding consists of only the (encodings of) circuits $C_1^{(i)}, \ldots, C_{3c}^{(i)}$ for all $1 \leq i \leq t$, which are precisely the sequence of circuits C (from the original promise axiom

$PRM_{C, \varepsilon \cdot 2^{3c}}$). Denote by Z this encoding. By Lemma 7 Item (b), there is a constant-size (independent of n) resolution refutation of K' and Z, and we are done. ∎$_{\text{Claim}}$

This concludes the proof of Lemma 8. □

Comment 3. We have defined our promise proof systems to be sound and complete for the set of CNF formulas under the promise (Definition 3.1). However, it is interesting to look at what happens if we consider "promise proofs" of formulas that are *satisfiable* but that have less than Λ (the promise) many satisfying truth assignments. Here we face a difficulty, since we are not always able to discard the *precise number* of assignments we would like to discard. That is, using the big promise axiom we can discard $\varepsilon \cdot 2^n$ assignments *only for ε a power of $\frac{1}{2}$* (in other words, we always discard the biggest power of 2 that does not exceed the promise). For example, if our promise is $\frac{1}{3} \cdot 2^n$ then the corresponding promise axiom discards only $\frac{1}{4} \cdot 2^n$ assignments. This means that we cannot treat our promise proof system as a sound proof system for all formulas (that is, even those that *are* satisfiable) having at most Λ satisfying assignments: if the promise $\varepsilon \cdot 2^n$ is not a power of 2, then we must discard less than $\varepsilon \cdot 2^n$ assignments, which means for instance that formulas with precisely $\varepsilon \cdot 2^n$ satisfying assignments will not be refutable even when having the promise axiom as an assumption in the refutation (since the conjunction of the promise axiom and the formula would clearly still be satisfiable).

It is also worth noting that there is another difficulty with refuting satisfiable formulas under a promise. The difficulty is that the polynomial-size refutation demonstrated above (in the proof of Theorem 4) for (unsatisfiable) 3CNF formulas under a big promise does not constitute a refutation when the 3CNF is satisfiable and has less than the promise satisfying assignments (even when the promise is a power of 2).[9]

5. Smaller Promise: Lower Bound

In this section, we prove an exponential lower bound on the size of resolution refutations under the promise $2^{\delta n}$, for any constant $0 \leq \delta \leq 1$. The lower bound applies to random 3CNF formulas with $O(n^{3/2-\epsilon})$ number of clauses, for $0 < \epsilon < \frac{1}{2}$ (where n is the number of variables in the 3CNF). This lower bound matches the known lower bound on resolution refutation-size for random 3CNF formula (without any promise). Basically, the proof strategy of our lower bound is similar to that of Ben-Sasson and Wigderson [BSW01], except that we need to take care that every step in the proof works with the augmented (smaller) promise axiom.

The lower bound is somewhat stronger than described above in two respects. First, we show that restricting the set of all truth assignments 2^n to *any* smaller set (that is, not just those sets defined by small circuits) that consists of $2^n - 2^{\delta n}$ assignments (for any constant $0 \leq \delta \leq 1$), does not give resolution any advantage in the average-case. One can think

---

[9]The reason for this is as follows. Assuming we are given a promise $\varepsilon \cdot 2^n$ which *is* a power of 2 and we are asked to refute in resolution under this promise a *satisfiable* CNF K having at most $\varepsilon \cdot 2^n$ satisfying assignments. Then, we cannot follow our refutation described in the proof of Theorem 4. For, assume that there is a set of at most $3(c-1)$ variables that hit all the clauses in K (as in case (i) in Lemma 5). Then we wish to continue in a similar manner as shown above in the proof of Lemma 6: that is, consider each of the constant number of possible assignments to the set of hitting variables, which then reduces K to a 2CNF. But now the problem is that when reducing K to a 2CNF we might still get a *satisfiable* formula (in contrast to the case in Lemma 6, where we assume from the start that K is unsatisfiable); thus, we cannot simply refute the resulted 2CNF using the known short refutations for 2CNFs as in [Coo71].

of such a restriction as modifying the semantic implication relation $\models$ to take into account only assignments from some prescribed set of assignments S, such that $|S| = 2^n - 2^{\delta n}$ (in other words, for two formulas A, B, we have that $A \models B$ under the restriction to S iff any truth assignment *from S* that satisfies A also satisfies B). Formally, this means that the lower bound does not use the fact that the restricted domain of size $2^n - 2^{\delta n}$ is defined by a sequence C of polynomial-size circuits (nor the fact that the circuits in C ought to have polynomial-size resolution proofs of their injectivity and pairwise disjointness).

Second, the lower bound rate can be improved as noted in Remark 1 below.

The following defines the usual *average-case* setting of 3CNF formulas (there are other definitions, that are essentially similar):

Definition 5.1 (Random 3CNF formulas)**.** For a 3CNF formula K with n variables X and $\beta \cdot n$ clauses, we say that β is the *density* of K. A *random* 3CNF *formula* on n variables and density β is defined by picking $\beta \cdot n$ clauses from the set of all $2^3 \cdot \binom{n}{3}$ clauses, independently and indistinguishably distributed, with repetitions.

We say that an event (usually a property of a 3CNF in n variables and density β) happens *with high probability* if it happens with $1 - o(n)$ probability in the specified probability space (usually random 3CNF formulas as defined in Definition 5.1).

Our goal is to prove a lower bound on the average-case refutation-size of 3CNF formulas taken from the *set of 3CNF formulas under a promise* as defined in Definition 3.1 (note that the probability space defined in Definition 5.1 is defined over a different set of 3CNF, that is, the set of *all* 3CNF formulas). For this purpose, we define a probability space over the set of 3CNF formulas under a promise: the distribution of *random* 3CNF *formulas under a promise* Λ on n variables and density β is the distribution of random 3CNF formulas in Definition 5.1 *conditioned* on the event that the 3CNF is either unsatisfiable or has more than $\Lambda(n)$ satisfying assignments.

We now argue that to satisfy our goal to prove a lower bound on the average-case proof complexity of 3CNF formulas under a promise, it is sufficient to prove the lower bound result considering the distribution of random 3CNF formulas as defined in Definition 5.1.

It is well known that almost all 3CNF formulas with a density β above a certain constant threshold (say, 5) are unsatisfiable. This means that any property of a 3CNF (with density above the threshold) that happens with high probability in the distribution in Definition 5.1 also happens with high probability in the distribution of random 3CNF formulas under a promise $\Lambda(n)$ (as defined above); this is because there are only a fraction $o(1)$ of 3CNF formulas (with a given fixed number of variables n and a given fixed density β above the threshold) that are satisfiable (and moreover have at least one satisfying assignment but less than $\Lambda(n)$ satisfying assignments). Thus, if we prove that with high probability a random 3CNF formula has no small promise resolution refutation then it implies also that with high probability *a random 3CNF formula under a promise* has no small promise resolution refutation. *Therefore, we shall consider from now on only the distribution of 3CNF formulas as defined in Definition 5.1, and forget about the other distribution.*

5.1. **The Lower Bound.** Throughout this section we fix $0 < \delta < 1$ and let $\Lambda = 2^{\delta n}$. For K a CNF formula, we denote by $\mathrm{Vars}(K)$ the set of variables that occur in K.

The following is the main theorem of this section. The lower bound matches that appearing in [BSW01] for resolution.

Theorem 9. *Let $0 < \delta < 1$ and $0 < \epsilon < 1/2$. With high probability a random $3CNF$ formula with $\beta = n^{1/2-\epsilon}$ requires a size $\exp(\Omega(\beta^{-4/(1-\epsilon)} \cdot n))$ resolution refutation under the promise $\Lambda = 2^{\delta n}$.*

Remark 1. We could allow in Theorem 9 for a promise that is bigger than $2^{\delta n}$, and precisely for a promise of $2^{n(1-\frac{1}{n^{1-\xi}})} = 2^n/2^{n^\xi}$, for any constant ξ such that $\frac{\epsilon}{(1-\epsilon)} < \xi < 1$ (for instance, this allows for a promise of $2^n/2^{n^{1/3}}$).

The actual proof of the lower bound uses the smaller promise of $2^{\delta n}$, but the proof for a $2^n/2^{n^\xi}$ promise is the same. (Although we have not defined precisely how the promise axioms are formulated in the case of a promise equal to $2^n/2^{n^\xi}$, it is possible to formulate such promise axioms along the same lines described in Definition 3.4.)

The proof strategy of Theorem 9 is to show that with high probability for a random 3CNF formula K with density $\beta = n^{1/2-\epsilon}$, a resolution refutation under the promise $2^{\delta n}$ of K must contain some clause D of large width (no matter which promise axioms $\mathrm{PRM}_{C,2^{\delta n}}$ we use). Then we can apply the size-width tradeoff from Theorem 1 to reach the appropriate size lower bound.

To illustrate an exponential lower bound via the size-width tradeoff of Theorem 1, we need to guarantee that all the initial clauses (that is, all the axiom clauses) are of *constant width*. The 3CNF formula K is clearly of constant width, but the clauses pertaining to the promise axiom $\mathrm{PRM}_{C,\Lambda}$ might not be (see the appendix for a detailed specification of these clauses). To solve this problem we add yet more extension variables to encode the clauses of the promise axiom with new constant width clauses. Second, we note that the original clauses of the promise axiom can be derived by a linear-size resolution proof from the new constant width version of the promise axiom (therefore, if there is a polynomial-size resolution refutation of K using the original promise axiom, then there is also a polynomial-size resolution refutation of K using the new constant width version of the promise axiom). Finally, we prove the exponential lower bound on resolution augmented with the constant width version of the promise axiom (instead of the original clauses pertaining to the promise axiom).

We now explain the standard way how to get the new constant-width promise axiom from the clauses pertaining to the original promise axiom from Definition 3.4 (as depicted in the appendix). Let $E = \ell_1 \vee \ldots \vee \ell_m$ be a clause in the promise axiom that has more than constant width (that is, ℓ_i's are literals and $m = \omega(1)$). Then, we replace the clause E with the following collection of clauses:

$$\ell_1 \vee e_1, \ \neg e_1 \vee \ell_2 \vee e_2, \ \neg e_2 \vee \ell_3 \vee e_3, \ \ldots, \neg e_{m-1} \vee \ell_m, \tag{7}$$

where the e_i's are new extension variables. By resolving on the e_i variables, one after the other, it is possible to derive with a linear-size resolution proof the original clause E from the clauses in (7) (consider the first two clauses (from left) in (7), and resolve over the variable e_1, then the resolvent of this step is resolved over the variable e_2 with the third clause in (7), and so forth). (Note that every truth assignment that satisfies (7) also satisfies E, and so any clause that is semantically implied by E (see the preliminaries, Section 2) is also semantically implied by (7). This means that the new constant width version of the promise axiom discards the same truth assignments to the variables X as the original version of the promise axiom.)

Thus, *from now on in this section we assume that the promise axiom consists of clauses of a constant width.*

The rest of this section is devoted to the proof of Theorem 9.

Let n be the number of (original, $x_1, \ldots, x_n$) variables, $\Lambda = 2^{\delta n}$ and let $\mathrm{PRM}_{C,\Lambda}$ be some given promise axiom (of constant width, independent of n). For a clause D define:

$$\eta(D) := \min\left\{ |K'| \;\middle|\; K' \subseteq K \text{ and } (\mathrm{PRM}_{C,\Lambda} \cup K') \models D \right\}.$$

Remark 2. We use the symbol η to distinguish it from a similar measure μ used in [BSW01]: here we require the minimal set of clauses from K that *combined with the axiom* $\mathrm{PRM}_{C,\Lambda}$ semantically imply D.

We show that with high probability for a random 3CNF formula with density $\beta = n^{1/2-\epsilon}$, for $0 < \epsilon < 1/2$, the following is true:

(1) Let $k = 2n \cdot (80\beta)^{-2/(1-\epsilon)}$. Then $\eta(\square) \geq k$.
(2) Any refutation of K must contain a clause D such that $k/2 \leq \eta(D) \leq k$.
(3) Any clause D from 2 must have large width, and specifically $|D| \geq \epsilon n (80\beta)^{-2/(1-\epsilon)}$ (which, by Theorem 1, concludes the proof).

The following two definitions are similar to those in [BSW01] (we refer directly to 3CNF formulas instead of 3-uniform hypergraphs):

Definition 5.2 (CNF Expansion). For a 3CNF formula K with density $\beta = n^{1/2-\epsilon}$, for $0 < \epsilon < 1/2$, the *expansion* of K is

$$e(K) := \min\left\{ 2|\mathrm{Vars}(K')| - 3|K'| \;\middle|\; \begin{array}{l} K' \subset K \text{ and} \\ n \cdot (80\beta)^{-2/(1-\epsilon)} \leq |K'| \\ \leq 2n \cdot (80\beta)^{-2/(1-\epsilon)} \end{array} \right\}.$$

Definition 5.3 (Partial matchability). A 3CNF formula K with density $\beta = n^{1/2-\epsilon}$, for $0 < \epsilon < 1/2$, is called *partially matchable* if for all $K' \subset K$ such that $|K'| \leq 2n \cdot (80\beta)^{-2/(1-\epsilon)}$ we have $|\mathrm{Vars}(K')| \geq |K'|$.

The next lemma gives two properties of random 3CNF formulas that occur with high probability (see the appendix of [BSW01] for a proof). We then use this lemma to show that with high probability 1,2,3 above hold.

Lemma 10 ([BKPS02]). *Let $0 < \epsilon < 1/2$ and let K be a random 3CNF with n variables and density $\beta = n^{1/2-\epsilon}$, then with high probability:*

(i) $e(K) \geq \epsilon n (80\beta)^{-2/(1-\epsilon)}$; *and*
(ii) K *is partially matchable.*

Proof of (1). In light of part (ii) in Lemma 10, in order to prove that with high probability (1) holds it is sufficient to prove the following:

Lemma 11. *Let K be a 3CNF formula in the X variables with density $\beta = n^{1/2-\epsilon}$, for $0 < \epsilon < 1/2$. If K is partially matchable then $\eta(\square) \geq 2n \cdot (80\beta)^{-2/(1-\epsilon)}$.*

Proof. By partial matchability of K, for all $K' \subset K$ such that $|K'| \leq 2n \cdot (80\beta)^{-2/(1-\epsilon)}$ it happens that $|\mathrm{Vars}(K')| \geq |K'|$. Thus, by Hall's Theorem we can choose a distinct variable (representative) from each clause in K' and set it to satisfy its clause. Clearly,

$|\text{Vars}(K')| \le 3|K'| \le 6n \cdot (80\beta)^{-2/(1-\epsilon)}$, and so there is a (partial) truth assignment ρ to at most $6n \cdot (80\beta)^{-2/(1-\epsilon)}$ variables in X that satisfies K'. Since $\beta = n^{1/2-\epsilon}$,

$$6n \cdot (80\beta)^{-2/(1-\epsilon)} = 6 \cdot 80^{-2/(1-\epsilon)} \cdot n^{\epsilon/(1-\epsilon)}, \tag{8}$$

which, by $0 < \epsilon < 1/2$, is equal to $O(n^\lambda)$ for some $0 < \lambda < 1$. Thus for sufficiently large n there are more than δn variables from X not set by ρ, which means that there are more than $2^{\delta n}$ different ways to extend ρ into truth assignments (to all the variables in X) that satisfy K'.[10] Since the promise axiom $\text{PRM}_{C,\Lambda}$ can discard up to $2^{\delta n}$ truth assignments to the X variables, we get that $\text{PRM}_{C,\Lambda} \cup K'$ is *satisfiable* (any assignment to X that is not discarded by $\text{PRM}_{C,\Lambda}$ can be extended to the extension variables in a way that satisfies $\text{PRM}_{C,\Lambda}$).

We have thus showed that every collection K' containing at most $2n \cdot (80\beta)^{-2/(1-\epsilon)}$ clauses from K and augmented with the promise axiom $\text{PRM}_{C,\Lambda}$ is satisfiable. This implies in particular that $\eta(\square) \ge 2n \cdot (80\beta)^{-2/(1-\epsilon)}$. $\qquad\square$

Proof of (2). Note that the resolution rule is *sub-additive* with respect to η in the sense that for all three clauses E, F, D such that D is a resolvent of E and F, it holds that

$$\eta(E) + \eta(F) \ge \eta(D).$$

We also clearly have that for every axiom clause E (either from K or from the promise axiom):

$$\eta(E) = 1.$$

Let

$$k = 2n \cdot (80\beta)^{-2/(1-\epsilon)}.$$

By Lemma 11, with high probability for a 3CNF formula K with density $\beta = n^{1/2-\epsilon}$ (for $0 < \epsilon < 1/2$) it happens that $\eta(\square) \ge k$. By sub-additivity of the resolution rule with respect to η, in any resolution refutation of K under the promise Λ, there ought to be some clause D such that

$$k/2 \le \eta(D) \le k.$$

Proof of (3). Let D be a clause such that $k/2 \le \eta(D) \le k$ from 2 and let K' be the (minimal) set of clauses from K for which $\text{PRM}_{C,\Lambda} \cup K' \models D$ and $k/2 \le |K'| \le k$. We shall prove that (with high probability for a random 3CNF) $|D| \ge \epsilon n(80\beta)^{-2/(1-\epsilon)}$. In light of Lemma 10 part (i), in order to prove this, it is sufficient to prove the next two lemmas.

Define $\partial K'$, called the *boundary of K'*, to be the set of variables in K' that occur only *once* in K' (in other words, each variable in $\partial K'$ appears only in one clause in K').

Lemma 12. $|\partial K'| \ge e(K)$.

Proof. Every variable not in $\partial K'$ must be covered by at least two distinct clauses in K', and so $|\text{Vars}(K')| \le |\partial K'| + \frac{1}{2} \cdot (3|K'| - |\partial K'|)$. Thus, we have $|\partial K'| \ge 2|\text{Vars}(K')| - 3|K'| \ge e(K)$ (where the last inequality is by Definition 5.2 and since $k/2 \le |K'| \le k$). $\qquad\square$

Lemma 13. $|D| \ge |\partial K'|$.

---

[10] Actually, for sufficiently large n there are more than $\Omega(n - n^{\epsilon/(1-\epsilon)})$ such variables, from which we can assume the bigger promise $\Lambda = 2^n/2^{n^\xi}$, for any $\frac{\epsilon}{(1-\epsilon)} < \xi < 1$, as noted in Remark 1.

Proof. Let $x_i \in \partial K'$, for some $1 \leq i \leq n$, and denote by K_i the (unique) clause from K' that contains x_i. Assume by a way of contradiction that x_i does not occur in D.

By minimality of K' with respect to η and D there exists an assignment α (here we treat α as a *total* truth assignment, that is, a truth assignment to both the X variables and the extension variables in the promise axiom) such that

$$(K' \setminus K_i)(\alpha) = 1 \quad \text{and} \quad D(\alpha) = 0 \tag{9}$$

(as otherwise $(K' \setminus K_i) \models D$ which clearly implies $\text{PRM}_{C,\Lambda} \cup (K' \setminus K_i) \models D$, which then contradicts the minimality of K' with respect to η and D).

By assumption, x_i occurs neither in $K' \setminus K_i$ nor in D. Hence, we can flip the value of α on x_i so that $K_i(\alpha) = 1$ while still keeping (9) true. We thus have:

$$K'(\alpha) = 1 \quad \text{and} \quad D(\alpha) = 0 \tag{10}$$

Since $|K'| \leq k$, we have that $|\text{Vars}(K')| \leq 3k = 6n \cdot (80\beta)^{-2/(1-\epsilon)}$ (recall that $|K'|$ is the number of *clauses* in K'). If $|D| \geq |\partial K'|$ we are done. Otherwise,

$$|\text{Vars}(K')| + |D| < |\text{Vars}(K')| + |\partial K'| \leq 2|\text{Vars}(K')| \leq 12n \cdot (80\beta)^{-2/(1-\epsilon)}.$$

Thus, similar to equation (8), for sufficiently large n, the total number of distinct variables in K' and D is at most $|\text{Vars}(K')| + |D| = O(n^\lambda)$, for some $0 < \lambda < 1$. This means that for sufficiently large n there are more than δn variables from X for which flipping the value of α on them still validates (10).[11] Hence, there are more than $2^{\delta n}$ distinct assignments to the X variables for which (10) holds.

The promise axiom $\text{PRM}_{C,\Lambda}$ discards at most $2^{\delta n}$ assignments to the X variables. This means that there are at most $2^{\delta n}$ assignments ρ to the X variables that falsify $\text{PRM}_{C,\Lambda}$ (that is, that every extension of ρ to all the extension variables falsifies $\text{PRM}_{C,\Lambda}$), while *all* other assignments ρ to the X variables have an extension (to all the the extension variables) that *satisfies* $\text{PRM}_{C,\Lambda}$. Thus, by the previous paragraph there ought to be at least one assignment ρ to the X variables that has an extension ρ' to the extension variables, such that

$$\text{PRM}_{C,\Lambda}(\rho') = 1, \, K'(\rho') = 1 \quad \text{and} \quad D(\rho') = 0, \tag{11}$$

which contradicts the assumption that $\text{PRM}_{C,\Lambda} \cup K' \models D$. $\qquad\qquad\square$

6. Conclusion

This paper establishes a new framework of propositional proof systems that are able to separate the unsatisfiable CNF formulas from the set of CNF formulas having many satisfying assignments. We were able to analyze the complexity of basic cases pertaining to such proof systems, such as the case of a big promise (a constant fraction of all truth assignments) and the average-case proof complexity of refutations under a smaller promise (that is, a promise of $2^{\delta n}$, for any constant $0 < \delta < 1$).

One question we have not addressed is what can be gained (if at all) when we augment a stronger proof system than resolution, like bounded-depth Frege proof system or Frege proof system, with the promise axioms (for a small promise like $2^{\delta n}$, as for a big promise already resolution can efficiently refute all unsatisfiable 3CNF formulas).

---

[11] Again, similar to what was noted in the proof of Lemma 11, for sufficiently large n there are actually more than $\Omega(n - n^{\epsilon/(1-\epsilon)})$ such variables.

Another question that arises is whether the fact that we require the Boolean circuits in the promise axioms to be *provably injective* and to *provably posses disjoint images* (that is, provably inside resolution) constitutes a real restriction. (Note that the lower bound for resolution under the promise $2^{\delta n}$ in Section 5 did not use at all these requirements.) In other words, we ask whether there is a sequence of circuits $C^{(1)}, \ldots, C^{(t)}$ for which adding the axiom $\vee_{i=1}^{t} C^{(i)}(\overline{W}) \equiv X$ (where the parameter t and the number of variables in m are taken from the smaller promise axiom 3.4) to resolution (or a stronger proof system) gives a super-polynomial speed-up for some contradictory family of formulas over standard resolution (or the stronger proof system); but that we cannot prove efficiently in resolution (or the stronger proof system) that $C^{(1)}, \ldots, C^{(t)}$ are injective or that they have pairwise disjoint images?

A different and more general task is to come up with other natural models of propositional proof systems that capture a "relaxed" notion of soundness. For instance, Pitassi [Pit06] suggested considering "approximate proofs" in the framework of algebraic proof systems.

Finally, we have not dealt directly in this paper with the promise $\Lambda = 2^n/poly(n)$, though it is most likely that a similar upper bound (with a similar proof) to that shown in Section 4 also holds for this promise (when the promise axiom is modified accordingly). In this respect it is worth mentioning that Krajíček [Kra07] observed that the work of Razborov and Rudich [RR97] implies the existence (under a cryptographic conjecture) of a Boolean function g with n^{δ} input bits (denoted by $y_1, \ldots, y_{n^{\delta}}$) and n output bits (denoted by $g_1(y_1, \ldots, y_{n^{\delta}}), \ldots, g_n(y_1, \ldots, y_{n^{\delta}})$), for any constant $0 < \delta < 1$, that has the following property: given any CNF formula K in n variables $x_1, \ldots, x_n$, substituting $g_1(y_1, \ldots, y_{n^{\delta}}), \ldots, g_n(y_1, \ldots, y_{n^{\delta}})$ for the original x_i variables in K yields a new CNF formula that is unsatisfiable only if K has at most $2^n/n^{\omega(1)}$ satisfying assignments. This means that *under the promise $2^n/poly(n)$ the substitution g is sound*: any unsatisfiable CNF formula (clearly) stays unsatisfiable after the substitution, while any CNF with more than $2^n/poly(n)$ satisfying assignments stays satisfiable after the substitution.

APPENDIX A. ENCODINGS

A.1. **Encoding of Boolean Circuits and Promise Axioms.** In this section we describe in detail how the promise axioms are encoded as CNF formulas. We require that resolution should be able to efficiently prove some basic facts about the encoded circuits.

A.1.1. *Boolean circuit encoding.* The following definition is similar to the *circuit encoding* defined in Alekhnovich *et al.* in [ABSRW04] (note that it deals with a *single output bit* Boolean circuit):

Definition A.1 (Encoding of Boolean circuits)**.** Let $C(\overline{W})$ be a Boolean circuit (with $\vee, \wedge$ as fan-in two gates and $\neg$ a fan-in one gate) and m input variables $\overline{W} := w_1, \ldots, w_m$ and a *single* output bit. For every gate v of the circuit C we introduce a special extension variable y_v. For input gates w_j ($1 \le j \le m$) we identify y_{w_j} with w_j. We denote by y^1 the literal y and by y^0 the literal $\neg y$. The CNF formula $\|C(\overline{W})\|$ consists of the following clauses:

(i) $y_{v_1}^{\bar{\epsilon}_1} \vee y_{v_2}^{\bar{\epsilon}_2} \vee y_v^{\pi_\circ(\epsilon_1, \epsilon_2)}$, where v is a $\circ \in \{\vee, \wedge\}$ gate in C and v_1, v_2 are the two input gates of v in C and $\langle \epsilon_1, \epsilon_2 \rangle$ is any vector in $\{0,1\}^2$ and $\pi_\circ$ is the truth table function of $\circ$ (and $\bar{0} = 1$, $\bar{1} = 0$);

(ii) $y_{v_1}^{\bar{\epsilon}_1} \vee y_v^{\pi_\neg(\epsilon_1)}$, where v is a $\neg$ gate in C and v_1 is the single input gates of v in C, and $\epsilon_1 \in \{0,1\}$ and $\pi_\neg$ is the truth table function of $\neg$.

We write $\|C(\overline{W})\|(y)$ to indicate explicitly that the output gate v of C is encoded by the extension variable y.

A.1.2. *Encoding of the promise axioms.* We now give a detailed description of how the promise axioms are encoded as CNF formulas. We consider only the big promise axiom (Definition 3.3), but the other variant (Definition 3.4) is similar. We encode the promise axioms in a bottom-up manner, encoding the sub-formulas separately, and then combining all of them together.

We assume that a Boolean circuit $C(\overline{W})$ with n output bits is encoded as n distinct circuits and we write $\|C(\overline{W})\|(\overline{Y})$ to indicate explicitly that the output gates $v_1, \ldots, v_n$ of C are encoded by the extension variables $y_1, \ldots, y_n$ (where $\overline{Y} := \{y_1, \ldots, y_n\}$). This means that $\|C(\overline{W})\|(\overline{Y})$ is the CNF formula $\wedge_{i=1}^{n} \|C_i(\overline{W})\|(y_i)$, where $C_i(\overline{W})$ is the circuit computing the ith output bit of $C(\overline{W})$ and y_i is the variable that encodes (see Definition A.1) the (single) output bit of $C_i(\overline{W})$. We also require that if the (function computed by the) circuit $C(\overline{W})$ does not depend[12], on some input bit w_i, then w_i does not occur in the encoding of $C(\overline{W})$.

Let $1 \le k \le t$ (where the parameter t is taken from Definition 3.3). We first encode as a CNF formula the negation of the following sub-formula from the promise axiom:

$$C^{(k)}(\overline{W}_1) \equiv C^{(k)}(\overline{W}_2) \rightarrow \overline{W}_1 \equiv \overline{W}_2.$$

We denote this CNF encoding by $\neg\text{INJ}_k$ (where INJ stands for *injective*).

Definition A.2 ($\neg\text{INJ}_k$). Let $1 \le k \le t$ and $m = n - r$ (all the parameters are taken from Definition 3.3). Let $\overline{W}_1 := \{w_1^{(1)}, \ldots, w_m^{(1)}\}$, $\overline{W}_2 := \{w_1^{(2)}, \ldots, w_m^{(2)}\}$, $\overline{Y}_k := \{y_1^{(k)}, \ldots, y_n^{(k)}\}$ and $Z_k := \{z_1^{(k)}, \ldots, z_n^{(k)}\}$ be sets of new *distinct* extension variables. The CNF formula $\neg INJ_k$ consists of the following set of clauses:

(1) $\|C^{(k)}(\overline{W}_1)\|(\overline{Y}_k);\ \|C^{(k)}(\overline{W}_2)\|(\overline{Z}_k)$ (expresses that $\overline{Y}_k, \overline{Z}_k$ are the output bits of $C^{(k)}(\overline{W}_1), C^{(k)}(\overline{W}_2)$, respectively);

(2) $\neg u_i \vee \neg y_i^{(k)} \vee z_i^{(k)};\ \neg u_i \vee y_i^{(k)} \vee \neg z_i^{(k)}$, for all $1 \le i \le n$ (expresses that u_i implies $y_i^{(k)} \equiv z_i^{(k)}$);

(3) $v_i \vee w_i^{(1)} \vee w_i^{(2)};\ v_i \vee \neg w_i^{(1)} \vee \neg w_i^{(2)}$; for al $1 \le i \le m$ (expresses that $\neg v_i$ implies $w_i^{(1)} \not\equiv w_i^{(2)}$);

(4) $u_1, \ldots, u_n$ (expresses that $\overline{Y} \equiv \overline{Z}$);

(5) $\neg v_1 \vee \ldots \vee \neg v_m$ (expresses that $\overline{W}_1 \not\equiv \overline{W}_2$);

For simplicity of writing we introduce the following notation: let ℓ be a literal and let A be a CNF formula. We denote by $\ell \bigvee A$ the set of clauses (that is, the CNF formula) that results by adding to each clause of A the literal ℓ.

We now encode as a CNF formula denoted by $\neg\text{INJ}$ the negation of

$$\bigwedge_{k=1}^{t} \left(C^{(k)}(\overline{W}_1) \equiv C^{(k)}(\overline{W}_2) \rightarrow \overline{W}_1 \equiv \overline{W}_2 \right).$$

Definition A.3 ($\neg\text{INJ}$). The *CNF* formula $\neg INJ$ consists of the following set of clauses:

(1) $\neg p_k \bigvee \neg\text{INJ}_k$ for al $1 \le k \le t$ (expresses that INJ_k implies $\neg p_k$);

---

[12]We say that a Boolean function f does not depend on an input bit w_i if for all input assignments α to f, flipping the truth value of w_i in α does not change the value of f.

(2) $p_1 \vee \ldots \vee p_t$ (expresses $\vee_{k=1}^{t} \neg \text{INJ}_k$.)

In a similar manner one can encode as a CNF the negation of the formula

$$\bigwedge_{1 \leq i < j \leq t} \left(C^{(i)}(\overline{W}_1) \not\equiv C^{(j)}(\overline{W}_2) \right),$$

denoted by $\neg\text{DSJ}$ (where DSJ stands for *disjoint*). We shall not develop the encoding precisely, as this is pretty much similar to $\neg\text{INJ}$.

The last part of the promise axiom we need to encode is the formula

$$\bigvee_{i=1}^{t} C^{(i)}(\overline{W}_1) \equiv X.$$

We denote the CNF encoding of this formula by RST (which stands for *restriction*). Again, this is similar to the encoding of $\neg\text{INJ}$, but we show how to encode it anyway, since we would like to illustrate in the sequel how resolution can use RST to efficiently prove some basic facts about the X variables (in the case the circuits in C have certain simple form).

Definition A.4 (RST). For every $1 \leq k \leq t$, recall that $\overline{Y}_k := \{y_1^{(k)}, \ldots, y_n^{(k)}\}$ are the output variables of $\|C^{(k)}(\overline{W}_1)\|$ from Definition A.2. The *CNF* formula *RST* consists of the following set of clauses:

(1) $\neg f_i^{(k)} \vee \neg y_i^{(k)} \vee x_i$; $\quad \neg f_i^{(k)} \vee y_i^{(k)} \vee \neg x_i$, for all $1 \leq i \leq n$ (expresses that $f_i^{(k)}$ implies $y_i^{(k)} \equiv x_i$);

(2) $\neg h_k \vee f_1^{(k)}, \ldots, \neg h_k \vee f_n^{(k)}$ (expresses that h_k implies $\overline{Y}_k \equiv X$);

(3) $h_1 \vee \ldots \vee h_t$ (expresses $\displaystyle\bigvee_{i=1}^{t} \overline{Y}_k \equiv X$.)

Finally, the promise axiom $\text{PRM}_{C,\Lambda}$ is the following CNF formula:

Definition A.5 (CNF encoding of $\text{PRM}_{C,\Lambda}$). The CNF encoding of the promise axiom $\text{PRM}_{C,\Lambda}$ consists of the following clauses:

(1) $\neg q_1 \ovee \neg\text{INJ}$ (expresses that q_1 implies $\neg\text{INJ}$);

(2) $\neg q_2 \ovee \neg\text{DSJ}$ (expresses that q_2 implies $\neg\text{DSJ}$);

(3) $q_1 \ovee (q_2 \ovee \text{RST})$ (expresses $\neg\text{INJ} \vee \neg\text{DSJ} \vee \text{RST}$, which is equivalent to $\text{INJ} \wedge \text{DSJ} \rightarrow \text{RST}$).

A.1.3. *Proving basic facts about encoded circuits inside resolution.* The following simple claim illustrates how one can reason inside resolution, and specifically can "eliminate implications" inside resolution. Consider, for instance, line (1) in $\text{PRM}_{C,\Lambda}$ (Definition A.5). This line is logically equivalent to $\text{INJ} \rightarrow \neg q_1$. Assume that we already know INJ (which formally means that we have a resolution refutation of $\neg\text{INJ}$). We would like to arrive inside resolution at $\neg q_1$. The following straightforward claim illustrates how to do this in resolution.

Claim 2. Let A be an unsatisfiable CNF formula with a resolution refutation of size s and let ℓ be any literal. Then there is a resolution proof of ℓ from $\ell \ovee A$ of size s.

Proof of claim: Assume that the resolution refutation of A is the sequence of clauses $A_1, \ldots, A_s$, where $A_s = \square$ (the empty clause). Then $\ell \vee A_1, \ldots, \ell \vee A_s$ is a resolution proof

of $\ell \vee \square = \ell$ from $\ell \bigvee A$ (we assume that ℓ is not in any A_i; or else, by the weakening rule, the claim also holds).■$_{\text{Claim}}$

Note that Claim 2 implies that if there is a refutation of $\neg$INJ of size s, then there is also a proof of $\neg q_1$ of size s, from line (1) in $\text{PRM}_{C,\Lambda}$ (Definition A.5).

We now illustrate how resolution can efficiently prove a certain simple fact about simple circuits. This is needed (among other efficient proofs of similar simple facts) in order to show the upper bound in Section 4 (and specifically, it is used in Claim 1). Other similar facts about the Boolean circuits constructed in Section 4 can be proved inside resolution in a similar manner.

For some $1 \leq k \leq t$, let $C^{(k)}$ be a circuit from a sequence of circuits C (as in the promise axioms), where m and n are the number of input and output variables of $C^{(k)}$, respectively. Assume that the ith output bit of $C^{(k)}$ computes the jth input bit w_j for some $1 \leq j \leq m$ and $1 \leq i \leq n$. We require that resolution can efficiently refute (the encoding via Definition A.2 of):

$$C^{(k)}(\overline{W}_1) \equiv C^{(k)}(\overline{W}_2) \wedge w_j^{(1)} \not\equiv w_j^{(2)}$$

(note that by assumption this is clearly a contradiction).

Since $C_i^{(k)}$ just computes the jth input bit w_j, then in fact we can assume that the encoding $\|C_i^{(k)}(\overline{W})\|(y_i)$ consists of only the single clause w_j (remember that by Definition A.1 we identify between the variable *encoding an input gate* with the input variable w_j itself; and here we know that w_j is also the output variable). Thus, by (2) in Definition A.2 we have that the output bit y_j of $C_i^{(k)}(\overline{W}_1)$ equals the output bit z_j of $C_i^{(k)}(\overline{W}_2)$, and y_j is actually $w_j^{(1)}$ and z_j is actually $w_j^{(2)}$. Therefore, by Definition A.2 (3), we can prove v_j. So, by one resolution rule applied to A.2 (5), we are left with $\vee_{i \in [m] \setminus \{j\}} v_i$.

Assume that all but a number d of the output bits of $C^{(k)}$ compute some (distinct) input bit w_j, for some $1 \leq j \leq m$ (this assumption corresponds to the circuits we build in Section 4). Then the process described in the previous paragraphs can be iterated for all such output bits of $C^{(k)}$, in order to cut off (that is, resolve over) all the v_j variables in clause (5) in Definition A.2, until we reach only a disjunction of d variables v_j instead of clause (5) in A.2. We are thus left with d circuits. (Therefore, [by assumptions made in Lemma 8] we can now refute with a size s resolution refutation the encoding of

$$C^{(k)}(\overline{W}_1) \equiv C^{(k)}(\overline{W}_2) \wedge \overline{W}_1 \not\equiv \overline{W}_2 \tag{12}$$

[if indeed the circuit $C^{(k)}$ computes an injective map, which means that (12) is unsatisfiable].)

A.1.4. *Comments on decoding the encoded promise axioms.* In order to assert that promise resolution is a Cook-Reckhow proof system (see the first paragraph in Section 1.1 for a definition) we need to make sure that a promise resolution refutation can be identified as such in polynomial-time. For this, one needs to be able to verify whether a given CNF is an instance of the promise axiom.

This can be done by "decoding" the CNF that encodes the promise axiom $\text{PRM}_{C,\Lambda}$ and then checking that each circuit in C has the right number of input and output bits. Here we illustrate how this can be achieved.

First, it is possible to identify which are the clauses pertaining to the promise axioms out of all the clauses in the refutation (for instance, these are clauses used as axioms that are not part of the clauses of the CNF meant to be refuted). Second, it is also possible to identify which are the clauses of the promise axiom that are part of the circuit encoding (that is, clauses in line 1 in Definition A.2). It is then possible to check that the circuits are legitimate ones and have the intended number of input and output variables (we omit the details).

ACKNOWLEDGMENTS

REFERENCES

[ABSRW04] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Pseudo-random generators in propositional proof complexity. *SIAM J. Comput.*, 34(1):67–88, 2004. (A preliminary version appeared in *Proceedings of the 41st Annual Symposium on Foundations of Computer Science* (Redondo Beach, CA, 2000)). A.1.1

[BKPS02] Paul Beame, Richard Karp, Toniann Pitassi, and Michael Saks. The efficiency of resolution and Davis-Putnam procedures. *SIAM J. Comput.*, 31(4):1048–1075, 2002. 1.1, 4, 10

[BS01] Eli Ben-Sasson. *Expansion in Proof Complexity*. PhD thesis, Hebrew University, Jerusalem, Israel, September 2001. 4

[BSW01] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, 2001. (A preliminary version appeared in *Proceedings of the 31th Annual ACM Symposium on the Theory of Computing* (Atlanta, GA, 1999)). 1.1, 4, 2.3, 1, 5, 5.1, 2, 5.1, 5.1

[Coo71] Stephen A. Cook. The complexity of theorem proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on the Theory of Computing*, pages 151–158. ACM, New York, 1971. 4, 4, 9

[CR79] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979. 1.1

[FKO06] Uriel Feige, Jeong Han Kim, and Eran Ofek. Witnesses for non-satisability of dense random 3CNF formulas. In *Proceedings of the IEEE 47th Annual Symposium on Foundations of Computer Science*, 2006. 1.1

[Hir98] Edward Hirsch. A fast deterministic algorithm for formulas that have many satisfying assignments. *Logic Journal of the IGPL*, 6(1):5971, 1998. 1.1, 1

[Kra07] Jan Krajíček. Substitutions into propositional tautologies. *Information Processing Letters*, 101:163–167, 2007. 6

[Kra07] Jan Krajíček. Personal communication, 2007. 2

[Pit06] Toniann Pitassi. Using hardness in proof complexity. Talk given in *New Directions in Proof Complexity*, an Isaac Newton institute workshop, Cambridge, April 2006. 6

[Rob65] J. Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23-41, January 1965. 1.1

[RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. System Sci.*, 55(1, part 1):24–35, 1997. (A Preliminary version appeared in *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing* (Montreal, PQ, 1994)). 6

[Tre04] Luca Trevisan. A note on approximate counting for k-DNF. In *Proc. 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2004)*, Lecture Notes in Computer Science, vol. 3122, Springer, pp. 417–426, 2004. 1.1, 4

School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel
  E-mail address: nachumd@tau.ac.il

Mathematical Institute, Academy of Sciences of the Czech Republic, Žitná 25, Prague, CZ-11567, the Czech Republic
  E-mail address: tzameret@math.cas.cz