Mathematical Structures in Computer Science (2020), **30**, pp. 736–751
doi:10.1017/S096012952000016X

CAMBRIDGE UNIVERSITY PRESS

PAPER

NAE-resolution: A new resolution refutation technique to prove not-all-equal unsatisfiability

Hans Kleine Büning[1], P. Wojciechowski[2] and K. Subramani[2]*

[1]Universität Paderborn, Paderborn, Germany and [2]West Virginia University, Morgantown, WV 26506, USA
*Corresponding author. Email: k.subramani@mail.wvu.edu

(Received 28 February 2018; revised 25 June 2020; accepted 2 July 2020; first published online 27 October 2020)

Abstract

In this paper, we analyze Boolean formulas in conjunctive normal form (CNF) from the perspective of read-once resolution (ROR) refutation schemes. A read-once (resolution) refutation is one in which each clause is used at most once. Derived clauses can be used as many times as they are deduced. However, clauses in the original formula can only be used as part of one derivation. It is well known that ROR is not complete; that is, there exist unsatisfiable formulas for which no ROR exists. Likewise, the problem of checking if a 3CNF formula has a read-once refutation is **NP-complete**. This paper is concerned with a variant of satisfiability called not-all-equal satisfiability (NAE-satisfiability). A CNF formula is NAE-satisfiable if it has a satisfying assignment in which at least one literal in each clause is set to **false**. It is well known that the problem of checking NAE-satisfiability is **NP-complete**. Clearly, the class of CNF formulas which are NAE-satisfiable is a proper subset of satisfiable CNF formulas. It follows that traditional resolution cannot always find a proof of NAE-unsatisfiability. Thus, traditional resolution is not a sound procedure for checking NAE-satisfiability. In this paper, we introduce a variant of resolution called NAE-resolution which is a sound and complete procedure for checking NAE-satisfiability in CNF formulas. The focus of this paper is on a variant of NAE-resolution called read-once NAE-resolution in which each clause (input or derived) can be part of at most one NAE-resolution step. Our principal result is that read-once NAE-resolution is a sound and complete procedure for 2CNF formulas. Furthermore, we provide an algorithm to determine the smallest such NAE-resolution in polynomial time. This is in stark contrast to the corresponding problem concerning 2CNF formulas and ROR refutations. We also show that the problem of checking whether a 3CNF formula has a read-once NAE-resolution is **NP-complete**.

Keywords: Read-once, NAE-SAT, refutation, optimal length refutation

1. Introduction

This paper is concerned with techniques for checking not-all-equal satisfiability (NAE-satisfiability) for propositional formulas in conjunctive normal form (CNF). We refer to the problem of checking NAE-satisfiability as the NAE-SAT problem. Briefly, the NAE-SAT problem is concerned with checking if a CNF formula has a satisfying assignment in which each clause has at least one literal set to **false**. It is well known that the NAE-SAT problem for 3CNF formulas (also called NAE-3SAT) is **NP-complete** (Schaefer, 1978). Indeed, the problem remains **NP-complete**, even when all the literals in each clause are positive. The problem can be solved in polynomial time, when there are at most two literals per clause (Moore and Mertens, 2011; Papadimitriou, 1994).

It is not hard to see that the class of CNF formulas which is NAE-satisfiable is a proper subset of CNF formulas which are satisfiable in the ordinary sense. Therefore, proof systems for satisfiability may not be sound for checking NAE-satisfiability. Indeed, this is the case with resolution refutation (Robinson, 1965), which is complete for NAE-satisfiability but not sound. In other words, if a resolution refutation exists for a CNF formula, then the formula is definitely NAE-unsatisfiable (since it is unsatisfiable). However, if a refutation does not exist for a formula, then it may still be NAE-unsatisfiable. In this paper, we design a new resolution scheme called NAE-resolution which is simultaneously sound and complete for the NAE-SAT problem in CNF formulas.

Propositional proof complexity is concerned with lengths of proofs (alternatively refutations) in propositional logic (Beame and Pitassi, 1998). In order to discuss lengths of proofs, it is vital that we have a concrete proof system in mind (Urquhart, 1995). Several proof systems have been discussed in the literature including Frege Systems, Extended Frege Systems, Resolution, and so on. The notion of proof length in various proof systems is discussed in Buss (1999). Observe that if it can be established that the length of *any* proof (refutation) of a contradiction must be exponential in the length of the input formula, then we have in fact separated the class **NP** from the class **coNP** (Cook and Reckhow, 1974). By any proof, we mean a proof in any proof system. For a comprehensive introduction to propositional proof complexity, we refer the interested reader to Urquhart (1995).

Even if we focus on a particular proof system, there exist several variants with different computational complexities. For instance, in case of resolution refutations, the commonly studied variants are tree-like refutations, dag-like refutations, and read-once refutations (Harrison, 2009). Read-once refutations are the simplest from the conceptual perspective, since each clause (original or derived) can be used at most once.

One of the interesting avenues of research in proof theory is the investigation of incomplete proof systems, i.e., proof systems which are not guaranteed to provide a refutation, even if the given formula is unsatisfiable. The idea behind the investigation of such weak systems is the hope that we can find proofs of unsatisfiability more efficiently (Iwama and Miyano, 1995). This paper focuses on a weak proof system called read-once resolution (ROR). It is well known that ROR is an incomplete proof system (Iwama and Miyano, 1995). Furthermore, even asking if an arbitrary unsatisfiable CNF formula has a read-once refutation is **NP-complete**. As discussed before, read-once refutation is not sound for the purpose of checking NAE-satisfiability. We design a variant of ROR called read-once NAE-resolution which is sound but not complete.

The investigations of this paper are concerned with properties of read-once NAE-resolutions when applied to the NAE-SAT problem in CNF formulas.

The principal contributions of this paper are as follows[1]:

1. A proof of existence of read-once NAE-resolution refutations for every NAE-unsatisfiable 2CNF formula. This result is particularly interesting since the problem of checking whether a 2CNF formula has an ROR is **NP-complete** (Kleine Büning et al., 2018).
2. The design and analysis of a polynomial time algorithm for finding read-once NAE-resolution refutations for NAE-unsatisfiable 2CNF formulas.
3. The design and analysis of a polynomial time algorithm for finding shortest read-once NAE-resolution refutations for NAE-unsatisfiable 2CNF formulas.
4. A proof that the algorithm for shortest read-once NAE-resolution refutations also finds shortest tree-like NAE-resolution refutations and shortest dag-like NAE-resolution refutations for NAE-unsatisfiable 2CNF formulas.
5. The design and analysis of a polynomial time algorithm for finding minimum weight read-once NAE-resolution refutations for NAE-unsatisfiable 2CNF formulas.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

6. A proof that the problem of checking if a 2CNF formula is minimal with respect to read-once NAE-resolution refutation can be solved in polynomial time.
7. A proof that the problem of checking if a 3CNF formula has a read-once NAE-resolution refutation is **NP-complete**.

The rest of this paper proceeds as follows: In Section 2, we cover some of the basic concepts necessary for our results. Section 3 formally defines the problems studied in this paper. In Section 4, we discuss the motivation for our work and examine related approaches in the literature. Sections 5 and 6 describe the results that we have obtained. Finally, in Section 7, we summarize our results and describe avenues of future research.

2. Preliminaries

In this section, we cover the concepts and terminology of propositional logic.

A formula Φ in CNF is a conjunction of clauses. Each clause in Φ is disjunction of literals written as $(L_1, \ldots, L_t)$. A literal is a propositional variable x or its negation, $\neg x$. Let $\phi = (L_1, \ldots, L_t)$ be a clause, then ϕ^c is the clause $(\neg L_1, \ldots, \neg L_t)$. Throughout this paper, we use n to denote the number of variables in Φ and m to denote the number of clauses in Φ.

We now recall some definitions with respect to NAE-satisfiability.

Definition 1. *A clause is* NAE-satisfied *by a truth assignment* **v**, *if at least one literal in the clause is assigned a value of* ***false*** *and at least one literal is assigned a value of* ***true****.*

A CNF formula, Φ, is *NAE-satisfiable* if there exists a truth assignment, **v**, such that every clause of Φ is NAE-satisfied. The class of NAE-satisfiable formulas is denoted as NAE-SAT. Note that, if **v** NAE-satisfies a formula in CNF, then so does $\neg$**v**. For a literal L_i, we use $v(L_i)$ to denote the value of L_i under truth assignment **v**.

We now compare NAE-satisfiability to regular satisfiability.

Lemma 1. *Let Φ be a formula in CNF. $\Phi \in$ NAE-SAT if and only if $\Phi \cup \Phi^c \in$ SAT, where $\Phi^c := \{(\neg L_1, \ldots, \neg L_t) : (L_1, \ldots, L_t) \in \Phi\}$.*

Proof. Let Φ be NAE-satisfied by the truth assignment, **v**. Thus, every clause ϕ_j of Φ contains a literal L_i and a literal L_k for which $v(L_i) =$ **true** and $v(L_k) =$ **false**. Hence, **v** satisfies both ϕ_j and ϕ_j^c. Thus, **v** satisfies every clause in $\Phi \cup \Phi^c$.

Let $\Phi \cup \Phi^c$ be satisfied by the truth assignment, **v**. Thus, every clause $\phi_j \in \Phi$ contains a literal L_i such that $v(L_i) =$ **true**. Similarly, ϕ_j^c contains a literal L_k such that $v(L_k) =$ **true**. By construction, ϕ_j contains the literal $\neg L_k$. Thus, under truth assignment **v**, ϕ_j contains both a true literal and a false literal. This means that **v** NAE-satisfies the clause ϕ_j. Since ϕ_j is an arbitrary clause of Φ, **v** NAE-satisfies Φ. □

Lemma 1 immediately leads to the observation that deciding whether a formula Φ is in NAE-SAT can be performed by means of resolution on $\Phi \cup \Phi^c$. Instead of adding the complementary clause ϕ_i^c in the beginning, we extend the resolution calculus with a new rule. This rule generates complementary clauses on demand.

We now define the inference rules for NAE-resolution.

Definition 2. *Let L_i and K_j be literals, and let x be a variable.* NAE-resolution *uses the following inference rules:*

1. *Resolution:* $(L_1, \ldots, L_t, x), (\neg x, K_1, \ldots, K_r) \vdash_{\overline{RES}}^{1} (L_1, \ldots, L_t, K_1, \ldots, K_r)$.
2. *NAE-extension:* $(L_1, \ldots, L_t) \vdash_{\overline{NAE\text{-}ext}} (\neg L_1, \ldots, \neg L_t)$.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

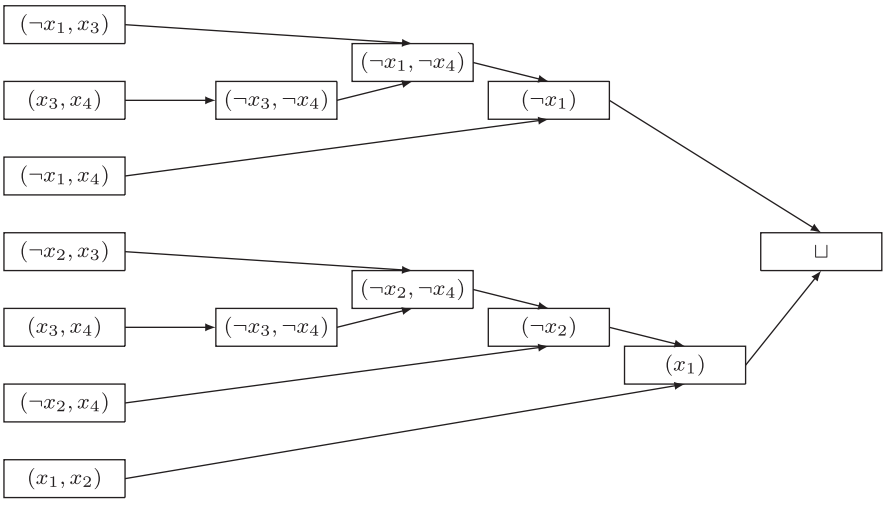

Figure 1. NAE-resolution derivation of the empty clause $\sqcup$.

NAE-resolution is an extension of resolution. In resolution, the only inference rule is the resolution rule. A sequence of resolutions that derives a clause ϕ from a CNF formula Φ is known as a resolution derivation of ϕ. We denote this as $\Phi \vdash_{\overline{RES}} \phi$. If ϕ is the empty clause $\sqcup$, then this is a resolution refutation of Φ.

Note that the NAE-extension rule is what allows us to simultaneously negate all the literals of a clause. If, given a CNF formula Φ, there is a sequence of resolutions and NAE-extensions that results in the clause ϕ, then we say that $\Phi \vdash_{\overline{NAE\text{-}Res}} \phi$. This sequence is called an NAE-resolution derivation of ϕ.

Example 1. Consider the 2CNF Formula (1):

$$(x_1, x_2)\ (\neg x_1, x_3)\ (\neg x_1, x_4)$$
$$(\neg x_2, x_3)\ (\neg x_2, x_4)\ (x_3, x_4) \tag{1}$$

This formula is satisfied by assigning $\mathbf{x} = (\mathbf{true}, \mathbf{true}, \mathbf{true}, \mathbf{true})$. However, this formula is not NAE-satisfiable. This can be seen in Figure 1.

It can easily be seen that NAE-resolution preserves NAE-satisfiability. That is, if the original formula is NAE-satisfiable, then any formula we get by adding the clauses introduced by resolution and the clauses introduced by the NAE-extension rule is NAE-satisfiable.

The following theorem summarizes the relationships between resolution and NAE-resolution. Note that throughout this paper we assume that no CNF formula Φ is trivially unsatisfiable. That is, we assume that $\sqcup \notin \Phi$.

Theorem 1. *Let Φ be a formula in CNF. The following propositions are equivalent:*

1. $\Phi \notin$ *NAE-SAT.*
2. $\Phi \cup \Phi^c \notin$ *SAT.*
3. $\Phi \cup \Phi^c \vdash_{\overline{RES}} \sqcup$.
4. *There exists some literal* L : $\Phi \vdash_{\overline{NAE\text{-}Res}} (L)$.
5. $\Phi \vdash_{\overline{NAE\text{-}Res}} \sqcup$.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

Proof. The proof of this is broken up as follows:

1. $\Phi \notin$ NAE-SAT if and only if $\Phi \cup \Phi^c \notin$ SAT:
   This was already proved in Lemma 1.
2. $\Phi \cup \Phi^c \notin$ SAT if and only if $\Phi \cup \Phi^c \vdash_{\overline{RES}} \sqcup$:
   Resolution is a sound and complete proof system (Tseitin, 1983). This result is an immediate consequence.
3. $\Phi \notin$ NAE-SAT if and only if there exists some literal $L : \Phi \vdash_{\overline{NAE\text{-}Res}} (L)$:
   See Theorem 2 for proof.
4. There exists some literal $L : \Phi \vdash_{\overline{NAE\text{-}Res}} (L)$ if and only if $\Phi \vdash_{\overline{NAE\text{-}Res}} \sqcup$:
   If $\Phi \vdash_{\overline{NAE\text{-}Res}} (L)$, then $\Phi \vdash_{\overline{NAE\text{-}Res}} (\neg L)$ since $(L) \vdash_{\overline{NAE\text{-}ext}} (\neg L)$. Thus, $\Phi \vdash_{\overline{NAE\text{-}Res}} \sqcup$ since $(L), (\neg L) \vdash_{\overline{RES}}^{1} \sqcup$.

   If $\Phi \vdash_{\overline{NAE\text{-}Res}} \sqcup$, then the final resolution step must be $(L), (\neg L) \vdash_{\overline{RES}}^{1} \sqcup$ for some literal L. Thus, $\Phi \vdash_{\overline{NAE\text{-}Res}} (L)$. □

Theorem 2. *A CNF formula Φ is not NAE-satisfiable if and only if for some variable x_i, $\Phi \vdash_{\overline{NAE\text{-}Res}} (x_i)$.*

Proof. Assume that for some x_i, $\Phi \vdash_{\overline{NAE\text{-}Res}} (x_i)$. We know that any assignment, **v**, that NAE-satisfies Φ must NAE-satisfy (x_i). However, the clause (x_i) has only one literal. Thus, it cannot be NAE-satisfied. This means that Φ is not NAE-satisfiable.

Let Φ be a CNF formula that is not NAE-satisfiable. We can construct the unsatisfiable formula $\Phi' = \Phi \cup \Phi^c$ of CNF clauses.

Since Φ' is unsatisfiable, we can derive the clauses (x_i) and $(\neg x_i)$ for some variable x_i. Let D be the read-once derivation $\Phi' \vdash_{\overline{ROR}} (x_i)$. Let $\phi_i \in \Phi^c$ be a clause used in D. Since $\phi_i \in \Phi^c$, $\phi_i^c \in \Phi$. Thus, ϕ_c can be derived from Φ by applying the NAE-extension rule to ϕ_i^c. This means that we can construct the NAE-resolution derivation $\Phi \vdash_{\overline{NAE\text{-}Res}} (x_i)$ from D as follows:

1. Let $\Phi_D \subseteq \Phi$ be the set of clauses in Φ used by D. Similarly, let $\Phi_D^c \subseteq \Phi^c$ be the set of clauses in Φ^c used by D.
2. For each clause $\phi_i \in \Phi_D^c$, $\phi_i^c \in \Phi_D$. Thus, we can derive ϕ_i by adding the NAE-extension step $\phi_i^c \vdash_{\overline{NAE\text{-}ext}} \phi_i$ to the NAE-resolution derivation of (x_i).
3. We have now derived all of the clauses in Φ' used by D. Thus, we can complete the NAE-resolution derivation of (x_i) by adding the resolution steps in D to the NAE-resolution derivation of (x_i).

Every resolution step and NAE-extension used in this derivation is applied to either clauses in Φ or to previously derived clauses. Thus, this is a valid NAE-resolution derivation of (x_i). □

From Theorem 2, for a CNF formula Φ, an NAE-resolution derivation $\Phi \vdash_{\overline{NAE\text{-}Res}} (x_i)$ is an NAE-resolution refutation of Φ.

3. Statement of Problems

In this section, we define the problems examined by this paper.

Let Φ be a CNF formula and let ϕ be a clause. An ROR derivation of ϕ, $\Phi \vdash_{\overline{ROR}} \phi$, is a resolution derivation, such that in each resolution step we remove the parent clauses from the current set of clauses and add the resolvent. A read-once refutation is a read-once derivation of the empty clause $\sqcup$.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

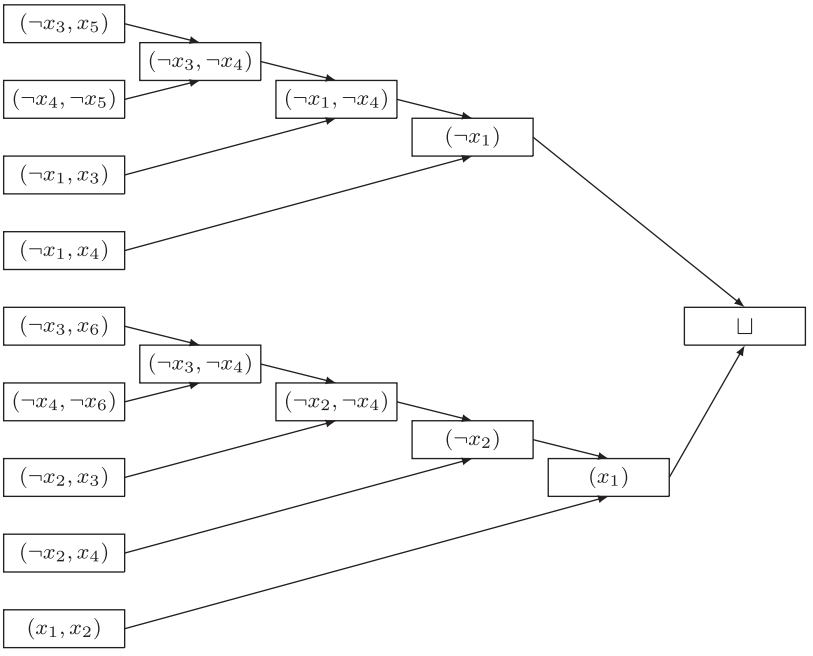

Figure 2. Read-once refutation.

Let ROR be the set of CNF formulas for which an ROR refutation exists. It has been shown (Iwama and Miyano, 1995) that ROR is **NP-complete**.

Example 2. We will generate an ROR refutation of the 2CNF formula specified by Formula (2):

$$\begin{array}{lll} (x_1, x_2) & (\neg x_1, x_3) & (\neg x_1, x_4) \\ (\neg x_2, x_3) & (\neg x_2, x_4) & (\neg x_3, x_5) \\ (\neg x_3, x_6) & (\neg x_4, \neg x_5) & (\neg x_4, \neg x_6) \end{array} \tag{2}$$

This can be seen in Figure 2.

Note that the clause $(\neg x_3, \neg x_4)$ is used twice. However, this is still a read-once refutation since each time the clause $(\neg x_3, \neg x_4)$ is derived, different clauses from the original formula are used.

Definition 3. *Let Φ be a CNF formula and let ϕ be a clause. A* read-once NAE-resolution derivation *of ϕ, $\Phi \vdash_{\overline{RO\text{-}NAE\text{-}Res}} \phi$, is a derivation using the resolution rule and/or NAE-extension rule. In the case of resolution, we delete the parent clauses from Φ and add the resolvent. In the case of the extension rule, $\phi' \vdash_{\overline{NAE\text{-}ext}} \phi'^c$, we remove the clause ϕ' from Φ and add ϕ'^c.*

There are two ways to check for the existence of a read-once proof of NAE-unsatisfiability. Given a CNF formula Φ, we can check if $\Phi \cup \Phi^c$ has an ROR refutation. Alternatively, we can ask whether Φ has a read-once NAE-resolution refutation (under the resolution and NAE-extension rules).

These methods of checking for read-once refutations correspond to the following sets of CNF formulas:

1. ROR-NAE := $\{\Phi \in CNF \mid \Phi \cup \Phi^c \vdash_{\overline{ROR}} \sqcup\}$.
2. RO-NAE-RES := $\{\Phi \in CNF \mid \exists L : \Phi \vdash_{\overline{RO\text{-}NAE\text{-}Res}} (L)\}$.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

In this paper, we study the problems of determining if certain forms of CNF formulas, specifically 2CNF and 3CNF formulas, belong to these classes.

We now define the length of an NAE-resolution refutation.

Definition 4. *The* length *of an NAE-resolution refutation is the number of resolution steps in the refutation.*

This allows us to define the problem of finding a shortest read-once NAE-resolution refutation.

4. Motivation and Related Work

In this section, we motivate our work and describe some related approaches in the literature. This paper focuses on proving the NAE-unsatisfiability of 2CNF and 3CNF formulas. This is closely related to the NAE-SAT problem which has many applications.

The NAE-SAT problem for monotone CNF is equivalent to the hypergraph bicolorability problem and the set splitting problem (Garey and Johnson, 1979; Porschen et al., 2014; Rödl and Siggers, 2006). Similarly, the NAE-SAT problem for monotone 2CNF is equivalent to the graph bicolorability problem (Subramani and Gu, 2011). Thus, an NAE-resolution refutation of a monotone 2CNF formula is a proof that the corresponding graph is not bipartite.

In addition to the focus on NAE-SAT, this paper also examines the problem of finding read-once refutations. Resolution-based refutation is a sound and complete proof system, that is, a formula has a resolution refutation if and only if it is infeasible (Tseitin, 1983). It is one of the weaker proof systems (Beame and Pitassi, 1997). There exist formulas for which the shortest resolution-based proofs are exponentially long. However, these same formulas can still have polynomially sized refutations in stronger proof systems. Despite this, resolution-based refutation is still widely used due to its relative simplicity (Beame and Pitassi, 1996).

Other more powerful refutation systems exist. These include Frege Proofs, Sequent Calculus, the Davis–Putnam Procedure, and Extended Frege Proofs (Beame, 2004). However, it was still difficult to derive an exponential lower bound on refutation length for resolution-based refutations. This lower bound on proof length was discovered when examining the pigeonhole principle. Haken (1985) showed that any proof of the pigeonhole principle requires a number of resolution steps exponential in the size of the input formula.

To the best of our knowledge, this is the first paper to examine the problem of finding read-once refutations for NAE-SAT. However, the ROR problem for SAT has been extensively researched. It was shown in Iwama and Miyano (1995) that for arbitrary CNF formulas, the problem of finding read-once refutations is **NP-complete**. Later papers extended this result by placing additional restrictions on the type of refutation.

One such restriction is the requirement that each resolution step in the read-once refutation uses a unit clause. Such a refutation is known as a read-once unit resolution refutation. It was shown in Kleine Büning and Zhao (2002) that the problem of identifying if a CNF formula has this restricted form of read-once refutation is **NP-complete**. It is also possible to further restrict read-once refutations by requiring that no literal is reused by the refutation. Such a refutation is called literal-once. In Szeider (2001), it was shown that the problem of finding literal-once resolution refutations for CNF formulas is **NP-complete**.

In addition to examining restricted forms of read-once refutations, the problem of finding read-once refutations for restricted forms of CNF has also been examined. In Kleine Büning et al. (2018), it was shown that the problem of finding read-once refutations remains **NP-complete** for 2CNF.

We focus on read-once refutations because, unlike unrestricted resolution refutations, a read-once refutation is guaranteed to be short. This is useful since an important problem in proof complexity is that of finding short refutations (Beame and Pitassi, 1998). Research into short

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

refutations attempts to separate the classes **NP** and **coNP** by proving that no proof system is guaranteed to produce short refutations for every infeasible instance of an **NP-complete** problem. For the SAT problem, this is done by looking at proof systems of different complexities and constructing lower bounds on proof length in each system (Urquhart, 1995). Despite the fact that ROR is an incomplete proof system, studying read-once refutations may give insights into what types of CNF formulas are guaranteed to have short refutations.

The problem of finding short refutations has also been studied for unrestricted resolution. Iwama (1997) examined the problem of finding the shortest resolution proofs of arbitrary 3CNF formulas. It was shown, in that paper, that the problem is **NP-complete**. The problem of finding a shortest resolution refutation was also examined for Horn formulas. In Alekhnovich et al. (1998), it was shown that one cannot linearly approximate the length of the shortest resolution refutation of a Horn formula, unless **P=NP**.

5. Read-Once NAE-Resolution Refutations for 2CNF Formulas

In this section, we examine the problem of finding read-once NAE-resolution refutations for 2CNF formulas.

For 2CNF formulas, it has been shown that the problem of checking whether the formula has an ROR refutation is **NP-complete** (Kleine Büning et al., 2018). Now the question arises whether that is still the case for read-once NAE-resolution refutations.

We first show that a 2CNF formula Φ has a read-once NAE-resolution refutation if and only if $\Phi \cup \Phi^c$ has an ROR refutation.

Theorem 3. *Let Φ be a 2CNF formula. $\Phi \in$ ROR-NAE if and only if $\Phi \in$ RO-NAE-RES.*

Proof. Let Φ be in ROR-NAE. Thus, there exists an ROR refutation $\Phi \cup \Phi^c \vdash_{ROR} \sqcup$. The final step of this refutation must be resolving a pair of one literal clauses to derive the empty clause. Thus, we must have that, for some literal L, $\Phi \cup \Phi^c \vdash_{ROR} (L)$. Let D be the shortest such resolution derivation. Thus, there is no literal L' which can be derived by a shorter ROR derivation.

By construction, every resolution step (except the last one) of D results in a two literal clause. Thus, we can restructure D so that each resolution step is of the form

$$(L, x_i), (\neg x_i, x_j) \vdash^1_{RES} (L, x_j)$$

Let $d_1, \ldots, d_r$ be the resolution steps in D. Let ϕ be a clause such that both ϕ and ϕ^c are used in D. Without loss of generality, we can assume that $\phi = (x_i, x_j)$, and that the restructured derivation uses ϕ before it uses ϕ^c. There are four cases we need to consider:

1. The resolution step d_s involving ϕ is $(L, \neg x_i), \phi \vdash^1_{RES} (L, x_j)$ and the resolution step d_t involving ϕ^c is $(L, x_i), \phi^c \vdash^1_{RES} (L, \neg x_j)$. Thus, the sequence $d_s, \ldots, d_t$ of resolution steps derives (L, x_i) from $(L, \neg x_i)$. For each resolution step d_h of the form $(L, x_k), (\neg x_k, x_l) \vdash^1_{RES} (L, x_l)$, let d'_h be the resolution step $(x_i, x_k), (\neg x_k, x_l) \vdash^1_{RES} (x_i, x_l)$. The sequence of resolution steps $d'_{s+1}, \ldots, d'_t$ derives (x_i). This contradicts our construction of D.
2. The resolution step involving ϕ is $(L, \neg x_j), \phi \vdash^1_{RES} (L, x_i)$ and the resolution step involving ϕ^c is $(L, x_j), \phi^c \vdash^1_{RES} (L, \neg x_i)$. Thus, there must be a sequence of resolution steps which produced (L, x_j) from $(L, \neg x_j)$. As before, this means that the set of clauses used in these resolution steps can derive x_j. This contradicts our construction of D.
3. The resolution step involving ϕ is $(L, \neg x_j), \phi \vdash^1_{RES} (L, x_i)$ and the resolution step involving ϕ^c is $(L, x_i), \phi^c \vdash^1_{RES} (L, \neg x_j)$. Thus, D derives (L, x_i) twice. By removing the sequence of resolution steps between these two derivations of (L, x_i), we produce a shorter derivation of (L). This contradicts our construction of D.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

4. The resolution step involving ϕ is $(L, \neg x_i), \phi \vdash_{RES}^{1} (L, x_j)$ and the resolution step involving ϕ^c is $(L, x_j), \phi^c \vdash_{RES}^{1} (L, \neg x_i)$. Thus, D derives (L, x_j) twice. By removing the sequence of resolution steps between these two derivations of (L, x_j), we produce a shorter derivation of (L). This contradicts our construction of D.

Thus, ϕ and ϕ^c cannot be both used in D.
We now have $\Phi \vdash_{NAE\text{-}Res} (L)$ as follows:

1. Apply the NAE-extension rule to every clause $\phi \in \Phi$ such that ϕ^c is used in D. Note, we are guaranteed that ϕ is not used in D.
2. Derive (L) using the same resolution steps as D.

Now let Φ be in RO-NAE-RES. For some literal L, there exists a read-once NAE-resolution derivation $\Phi \vdash_{NAE\text{-}Res} (L)$. Let $\phi_1, \ldots, \phi_t$ be the set of clauses used in this read-once derivation. We can use the dual clauses $\phi_1^c, \ldots, \phi_t^c$ to derive $(\neg L)$. Note that if the NAE-extension rule is used in the original refutation, for example, as $\phi_i \vdash_{NAE\text{-}ext} \phi_i^c$, then we include ϕ_i^c in the read-once derivation of (L) and ϕ_i in the read-once derivation of $(\neg L)$.

As a final step, we can use (L) and $(\neg L)$ to derive the empty clause. This forms an ROR refutation of $\Phi \cup \Phi^c$. □

Theorem 4. *Let Φ be a 2CNF formula. We have $\Phi \notin$ NAE-SAT if and only if $\Phi \in$ ROR-NAE, and a read-once NAE-resolution refutation can be found in quadratic time.*

Proof. Let Φ be a 2CNF formula that is not in NAE-SAT. If Φ contains a unit clause, say (x), then $\{(x), (\neg x)\} \subseteq \Phi \cup \Phi^c$. We have that $(x), (\neg x) \vdash_{RES}^{1} \sqcup$. This is clearly a read-once NAE-resolution refutation. Thus, we assume that Φ contains no unit clauses.

From $\Phi \cup \Phi^c$, we create the implication graph, G, as described in Aspvall et al. (1979). This construction proceeds as follows:

1. For every variable x_i, we create the vertices x_i and $\neg x_i$.
2. For every clause $(L \vee K)$, we create the edges $\neg L \rightarrow K$ and $\neg K \rightarrow L$.

G contains a strongly connected component, say G_1, with a pair of vertices corresponding to complementary literals if and only if $\Phi \cup \Phi^c$ is unsatisfiable (Aspvall et al., 1979). From Theorem 1, a formula Φ is not in NAE-SAT if and only if $\Phi \cup \Phi^c$ is unsatisfiable. Thus, there exists a strongly connected component C in G that contains vertices corresponding to complementary literals if and only if $\Phi \notin$ NAE-SAT.

Let $\neg L_0 \rightarrow L_1 \rightarrow L_2 \ldots L_k \rightarrow L_0$ be a shortest path in C between vertices corresponding to a complementary pair of literals.

If, for some $1 \leq i < j \leq k$, we have $L_i = L_j$, then the path from $\neg L_0$ to L_0 could be shortened by removing the sub-path from L_i to $L_j = L_i$. Similarly, if $L_i = \neg L_j$, then the path from L_i to $L_j = \neg L_i$ would be a shorter path in C between vertices corresponding to a complimentary pair of literals. Since $\neg L_0 \rightarrow L_1 \rightarrow L_2 \ldots L_k \rightarrow L_0$ is a shortest path in C between such a pair of vertices, we know that neither of these situations can occur.

Let R_0 be the following resolution derivation:

$$(L_0 \vee L_1), (\neg L_1 \vee L_2), \ldots, (\neg L_m \vee L_0) \vdash_{ROR} (L_0)$$

We know that for every $1 \leq i < j \leq k$, $L_i \neq L_j$ and $L_i \neq \neg L_j$. Thus, no clause is used multiple times by R_0. Consequently, R_0 is a read-once derivation of (L_0).

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

Note that R_0 is a resolution derivation of (L_0) from $\Phi \cup \Phi^c$. Thus, the clauses $(\neg L_0 \vee \neg L_1)$, $(L_1 \vee \neg L_2), \ldots, (L_m \vee \neg L_0)$ are also in $\Phi \cup \Phi^c$. These clauses form an ROR derivation R_0^c of $(\neg L_0)$. Recall that for every $1 \leq i < j \leq k$, $L_i \neq L_j$ and $L_i \neq \neg L_j$. Thus, no clause is used in both R_0 and R_0^c. This means that $\Phi \cup \Phi^c$ has the following ROR refutation:

1. Use R_0 to derive the clause (L_0).
2. Use R_0^c to derive the clause $(\neg L_0)$.
3. Resolve (L_0) and $(\neg L_0)$ to get $\sqcup$.

By Theorem 3, this corresponds to a read-once NAE-resolution refutation of Φ.

The computation of the strongly connected components of G includes deciding whether a path exists between vertices corresponding to a complementary pair of literals. This costs linear time. Finding the shortest path between the vertices corresponding to a pair of complimentary literals takes linear time. Since we do this for each such pair in a strongly connected component of **G**, the overall running time of the algorithm is quadratic. □

Note that if we do not want the shortest path, then this procedure can be shortened to run in near-linear time by using union-find data structures (Cormen et al., 2009). This new procedure is described in the following:

1. From $\Phi \cup \Phi^c$, construct the implication graph **G**.
2. Find the connected components of **G**.
3. Each time two vertices are determined to be in the same component of **G**, add them into a union-find data structure. This can be done in $O(m \cdot \alpha(n))$ time where $\alpha(n)$ is the inverse Ackermann function (Cormen et al., 2009).
4. For each variable x_i in Φ check to see if x_i and $\neg x_i$ belong to the same component of **G**. Using the union-find data structure created before, this can be done in $O(n \cdot \alpha(n))$ time.
5. Find a path p from x_i to $\neg x_i$ in **G**.
6. Find the shortest sub-path of p between a pair of vertices corresponding to complementary literals. This can be done in $O(n)$ time.

Overall, this procedure runs in time $O((m + n) \cdot \alpha(n))$.

5.1 Finding shortest refutations

Earlier in Section 5, we described an implication graph for checking the satisfiability of 2CNF formulas. We can construct a similar implication graph for checking the NAE-satisfiability of 2CNF formulas. We refer to this as the NAE-implication graph. The NAE-implication graph of a formula Φ is equivalent to the implication graph of $\Phi \cup \Phi^c$

Example 3. Consider the 2CNF formula:

$$(x_1, x_2)\ (x_2, x_3)\ (\neg x_3, x_4)$$

From this formula, we can generate the NAE-implication graph in Figure 3.

We now show that, in the case of NAE-unsatisfiable 2CNF formulas, there always exists a read-once NAE-resolution refutation.

Theorem 5. *If a 2CNF formula, Φ, has an NAE-resolution derivation of (x_i), then it has a read-once NAE-resolution derivation of (x_i).*

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

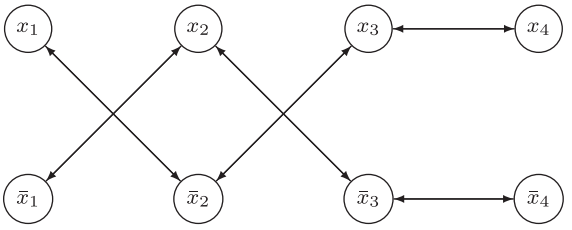

Figure 3. Example NAE-implication graph.

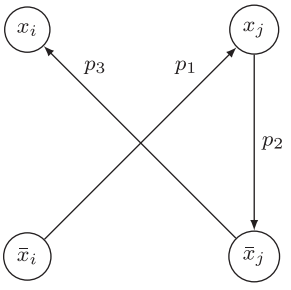

Figure 4. Example of path p.

Proof. Note that if Φ contains a unit clause (x_i), then this unit clause by itself is a proof of NAE-unsatisfiability. Thus, we can assume without loss of generality that Φ contains no unit clauses.

Let G be the NAE-implication graph corresponding to Φ. Note that **G** is also the implication graph of $\Phi \cup \Phi^c$. Thus, $\Phi \cup \Phi^c \vdash_{\overline{RES}} (x_i)$ if and only if there exists a path from $\bar{x}_i$ to x_i in G. From Theorem 2, $\Phi \vdash_{\overline{NAE\text{-}Res}} (x_i)$ if and only if $\Phi \cup \Phi^c \vdash_{\overline{RES}} (x_i)$. Thus, $\Phi \vdash_{\overline{NAE\text{-}Res}} (x_i)$ if and only if there exists a path from $\bar{x}_i$ to x_i in G. Note that the clauses used by the refutation correspond to the clauses used to form the path in the implication graph. Let p denote this path.

Let x_j be the first variable on the path p such that $\bar{x}_j$ also appears on p. This x_j is guaranteed to exist since both x_i and $\bar{x}_i$ appear on p. We can assume without loss of generality that x_j appears before $\bar{x}_j$. Thus, we can break p up as follows:

1. a path, p_1, from $\bar{x}_i$ to x_j,
2. a path, p_2, from x_j to $\bar{x}_j$,
2. and a path, p_3, from $\bar{x}_j$ to x_i.

This can be seen in Figure 4.

By our choice of x_j, we know that for $k \neq j$, p_1 and p_2 together do not contain both x_k and $\bar{x}_k$. As a consequence of this, no two edges in p_1 or p_2 correspond to the same constraint. Thus, p_2 corresponds to a read-once NAE-resolution derivation of $(\neg x_j)$. We also have that p_1 is a read-once NAE-resolution derivation of (x_i, x_j) which has no clauses in common with the NAE-resolution derivation corresponding to p_2. Combining these two yields a read-once NAE-resolution derivation of (x_i). □

From this theorem, read-once NAE-resolution is a complete proof system for NAE-2SAT.

Note that, in this NAE-resolution derivation, the sub-path p_2 from x_j to $\bar{x}_j$ is a proof of NAE-unsatisfiability by itself since it derives $(\neg x_j)$ which is already enough to force x_j to be both **true** and **false**.

This provides us with a polynomial time algorithm to find the smallest read-once NAE-resolution refutation of a 2CNF formula.

Note that we do not need to consider the paths from x_i to $\bar{x}_i$ since the existence of such a path means that there is a path of equal length from $\bar{x}_i$ to x_i.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

Algorithm 1 Algorithm for finding a minimum length read-once NAE-resolution refutation

Function FIND-MIN-READONCE-NAE-RESOLUTION-REFUTATION (Φ)

Input: Φ – An NAE-unsatisfiable formula with m clauses over n variables.

Output: $p_{\min}$ – A shortest read-once NAE-resolution refutation of Φ.

Use Φ to construct the NAE-implication graph G.
Let $l_{\min}$ denote the length of the shortest known refutation of Φ.
Initialize $l_{\min}$ to $(m+1)$.
▷ No read-once NAE-resolution refutation is longer than m.
Let $p_{\min}$ denote the shortest read-once NAE-resolution refutation of Φ.
Initialize $p_{\min}$ to $\emptyset$.
for $(i = 1$ **to** $n)$ **do**
    if (x_i is reachable from $\bar{x}_i$ in G) **then**
        Let p be the refutation corresponding to a shortest path from $\bar{x}_i$ to x_i in G.
        ▷ There is a path of equal length from x_i to $\bar{x}_i$.
        Let l be the length of p.
        if $(l < l_{\min})$ **then**
            $l_{\min} := l$ and $p_{\min} := p$.
return $(p_{\min})$.

This algorithm works when the length of a proof is determined solely by the number of resolution steps (see Definition 4) and does not depend on the number of NAE-extensions. If we include the number of NAE-extensions in the length of the proof, then we need to weigh the edges in the graph. Edges corresponding to clauses in Φ will have weight one and edges corresponding to clauses in Φ^c will have weight 2 since these edges need to be derived using the NAE-extension rule.

Since every NAE-unsatisfiable system has a read-once refutation, Algorithm 1 also returns a shortest tree-like and dag-like NAE-resolution refutations.

Theorem 6. *Algorithm 1 returns a shortest tree-like NAE-resolution refutation and a shortest dag-like NAE-resolution refutation of a system Φ.*

Proof. We will prove this for tree-like NAE-resolution refutations since the proof in the case of dag-like NAE-resolution refutations is analogous.

Let k_{ROR} be the length of the NAE-resolution refutation returned by Algorithm 1. Assume that Φ has a tree-like NAE-resolution refutation of length $(k_{tree} < k_{ROR})$. Let $\Phi' \subseteq \Phi$ be the set of constraints used by this tree-like NAE-resolution refutation. From Theorem 5, Φ' has a read-once NAE-resolution refutation. Let k'_{ROR} denote the length of this read-once NAE-resolution refutation. This refutation is also a read-once NAE-resolution refutation of Φ. However, $(k'_{ROR} \leq k_{tree} < k_{ROR})$. This contradicts the fact that Algorithm 1 returns a shortest read-once NAE-resolution refutation. Thus, since the NAE-resolution refutation returned by Algorithm 1 is also a tree-like NAE-resolution refutation, Algorithm 1 returns a shortest such refutation. □

Example 4. Consider the following system of constraints:

$$(\neg x_1, x_4) \quad (x_1, x_2) \quad (x_2, x_3) \quad (x_4, x_1)$$

This set of constraints has a Dag-like NAE-resolution refutation as shown in Figure 5.

However, the shortest read-once NAE-resolution refutation of this system is $(\neg x_1, x_4), (x_4, x_1) \vdash_{\overline{RES}}^{1} (x_4)$. Since this is a Dag-like refutation, it is a shorter Dag-like refutation than the one shown in Figure 5.

Algorithm 1 can be easily modified to solve the following problem.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

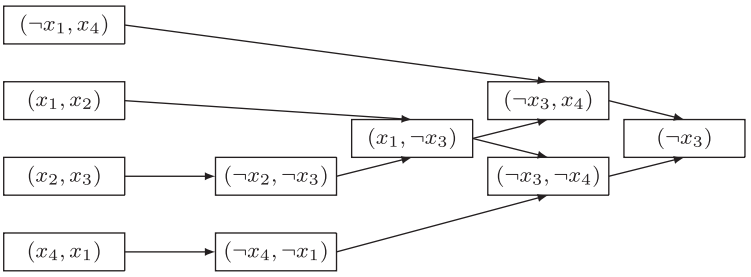

Figure 5. Dag-like NAE-resolution refutation.

Definition 5. *In the* minimum-weight *read-once NAE-resolution refutation problem, each clause of Φ is assigned a nonnegative weight. The goal is to find a read-once NAE-resolution refutation with minimum total weight.*

To find the minimum-weight read-once NAE-resolution refutation for a 2CNF formula, we construct a weighted NAE-implication graph. In this graph, each edge is assigned the same weight as the corresponding 2CNF clause. We then run a modified version of Algorithm 1 on this weighted graph to find the minimum-weight path from $\bar{x}_i$ to x_i.

We now discuss the notion of minimal NAE-read-once in CNF formulas.

Definition 6. *A CNF formula Φ is* minimal NAE-read-once*, if Φ has a read-once NAE-resolution refutation, but no sub-formula of Φ has a read-once NAE-resolution refutation.*

Definition 6 lets us define the following problem.

Definition 7. *The* Minimal NAE-resolution Refutability (MNRR) *problem is the problem of determining if a CNF formula Φ is minimal NAE-read-once.*

The computational complexity of the MNRR problem for general CNF formulas is unknown. However, Algorithm 1 can be used to solve the MNRR problem for 2CNF formulas in polynomial time.

Theorem 7. *MNRR for 2CNF formulas is in* **P**.

Proof. Let Φ be an NAE-unsatisfiable 2CNF formula, and let p be the path returned by running Algorithm 1 on Φ. If Φ is minimally NAE-read-once, then any read-once NAE-resolution refutation of Φ must use every clause in Φ. Thus p must use every clause in Φ.

As described above, the path p produced by Algorithm 1 is the minimum read-once NAE-resolution refutation of Φ. Thus, if p uses all the clauses of Φ, then any read-once NAE-resolution refutation of Φ must use all the clauses of Φ. This means that Φ is minimally NAE-read-once.

Since Algorithm 1 runs in polynomial time, the MNRR problem for 2CNF formulas is in **P**. □

6. Read-Once NAE-Resolution Refutations for 3CNF Formulas

In this section, we examine the problem of finding read-once NAE-resolution refutations for 3CNF formulas. We refer to this problem as ROR-NAE-3SAT.

Now we focus on applying NAE-resolution to formulas in 3CNF. We show that, given a formula Φ, the problem of checking whether the formula $\Phi \cup \Phi^c$ has an ROR refutation is **NP-complete**. Since ROR – the set of formulas in CNF for which an ROR exists – is **NP-complete**, we see that ROR-NAE-3SAT is in **NP**. Therefore, we only have to show **NP-hardness**. This is done

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

by a reduction to the problem of deciding whether a formula in 2CNF has an ROR refutation (ROR-2CNF). This problem is known to be **NP-complete** (Kleine Büning et al., 2018).

Let Φ be a 2CNF formula. We construct the 3CNF formula Φ^* as follows:

1. For each variable x_i of Φ, create the variable x_i for Φ^*.
2. Create the variable x_0 for Φ^*.
3. For each clause $\phi \in \Phi$, create the clause $(\phi \vee x_0) \in \Phi^*$.

Example 5. Consider the following 2CNF formula Φ:

$$(x_1, x_2) \quad (\neg x_2, x_3)$$
$$(\neg x_1, \neg x_4) \quad (x_3, \neg x_4)$$

Applying this construction to Φ results in the following 3CNF formula:

$$(x_1, x_2, x_0) \quad (\neg x_2, x_3, x_0)$$
$$(\neg x_1, \neg x_4, x_0) \quad (x_3, \neg x_4, x_0)$$

This construction is used to prove the **NP-completeness** of ROR-NAE-3SAT.

Theorem 8. *ROR-NAE-3SAT is **NP-complete**.*

Proof. Let Φ be a 2CNF formula and let Φ^* be the 3CNF formula made from Φ by the preceding construction. We show that $\Phi \in$ ROR-2CNF if and only if $\Phi^* \in$ ROR-NAE-3SAT.

Assume that $\Phi \in$ ROR-2CNF. An ROR refutation $\Phi \vdash_{\overline{ROR}} \sqcup$ can easily be extended to the read-once NAE-resolution derivation $\Phi^* \vdash_{\overline{ROR}} x_0$. Thus, by Theorem 2, Φ^* is in ROR-NAE-3SAT.

Now suppose that Φ^* is in ROR-NAE-3SAT. We must show that Φ has an ROR refutation. We do this by showing that every resolution step done on the 3CNF clauses corresponds to a valid derivation on the 2CNF clauses.

We have the following cases:

1. $(x_i, x_j, x_0) \vdash_{\overline{NAE\text{-}ext}} (\neg x_i, \neg x_j, \neg x_0)$: Both of these clauses correspond to the 2CNF clause (x_i, x_j). If (x_i, x_j) is satisfied, then both (x_i, x_j, x_0) and $(\neg x_i, \neg x_j, \neg x_0)$ are NAE-satisfied by setting x_0 to **false**.
2. $(x_i, x_j, x_0), (\neg x_k, \neg x_l, \neg x_0) \vdash^1_{\overline{RES}} (x_i, x_j, \neg x_k, \neg x_l)$: This corresponds to the two CNF clauses $(x_i, x_j, \neg x_k, \neg x_l)$ and $(\neg x_i, \neg x_j, x_k, x_l)$. However, these are made redundant by the 2CNF clauses (x_i, x_j) and (x_k, x_l) which are already derivable from Φ. Thus, no NAE-resolution refutation of Φ^* performs a resolution step centered on x_0.
3. $(x_i, x_j, x_0), (\neg x_j, \neg x_k, x_0) \vdash^1_{\overline{RES}} (x_i, \neg x_k, x_0)$: This corresponds to the resolution step $(x_i, x_j), (\neg x_j, \neg x_k) \vdash^1_{\overline{RES}} (x_i, \neg x_k)$. Since $\Phi \vdash_{\overline{RES}} (x_i, x_j)$ and $\Phi \vdash_{\overline{RES}} (\neg x_j, \neg x_k)$, this is a valid derivation from Φ.

Thus, all steps in the NAE-resolution refutation of the 3CNF formula correspond to steps used in the resolution refutation of the original 2CNF formula. Thus, Φ^* has a read-once NAE-resolution refutation if and only if Φ the has an ROR refutation.

The problem of finding read-one resolution refutations is **NP-complete** for 2CNF formulas (Kleine Büning et al., 2018). Thus, the problem of finding read-once NAE-resolution refutation for 3CNF formulas is **NP-complete**. □

Since there exist 2CNF formulas without read-once refutations, there exist 3CNF formulas instances without read-once NAE-resolution refutations.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

Example 6. The following 2CNF formula does not have a read-once refutation:

$$(x_1, x_2)\ \ (x_3, x_4)$$
$$(\neg x_1, \neg x_3)\ \ (\neg x_1, \neg x_4)$$
$$(\neg x_2, \neg x_3)\ \ (\neg x_2, \neg x_4)$$

Thus, by the preceding theorem, the following 3CNF formula does not have a read-once NAE-resolution refutation.

$$(x_1, x_2, x_0)\ \ (x_3, x_4, x_0)$$
$$(\neg x_1, \neg x_3, x_0)\ \ (\neg x_1, \neg x_4, x_0)$$
$$(\neg x_2, \neg x_3, x_0)\ \ (\neg x_2, \neg x_4, x_0)$$

7. Conclusion

In this paper, we introduced the notion of NAE-resolutions and show how they can be applied to the problem of checking NAE-satisfiability in CNF formulas. Prior to our work, the standard approach in the literature was to convert the NAE-satisfiability problem to simple satisfiability. Our principal contribution is showing that the problem of checking whether a 2CNF formula has a read-once NAE-resolution is in **P**. Furthermore, we showed that the problem of finding the optimal length read-once NAE-resolution is also in **P**. This is in stark contrast to the problem of checking whether a 2CNF formula has a read-once refutation, which we have shown to be **NP-complete** Kleine Büning et al. (2018).

From our perspective, the following problems are worth investigating:

1. We showed that the problem of checking whether a 3CNF formula has a read-once NAE-resolution refutation is **NP-complete**. An interesting related problem is the following: Given an NAE-unsatisfiable 3CNF formula which has at least one read-once NAE-resolution, what is the complexity of determining the optimal length read-once NAE-resolution?
2. We examined the problem of finding read-once NAE-resolution refutations for both 2CNF formulas and 3CNF formulas. However, the same problem remains open for other restricted CNF formulas including Horn formulas. It would be instructive to study the complexity of finding read-once NAE-resolution refutations of these other restricted forms of CNF formula.

Acknowledgements. This research was supported in part by the Air-Force Research Laboratory, Rome, through Contract FA8750-17-S-7007 and in part by the Air-Force Office of Scientific Research through grant FA9550-19-1-0177. We would like to thank the anonymous reviewers of our paper for their suggestions; in particular, one of the reviewers suggested an improved algorithm for checking if a 2CNF formula has a read-once NAE-resolution refutation.

Note

1 This paper extends the work in Kleine Büning et al. (2017) with additional results.

References

Alekhnovich, M., Buss, S., Moran, S. and Pitassi, T. (1998). Minimum propositional proof length is NP-hard to linearly approximate. In: *Mathematical Foundations of Computer Science (MFCS)*, Lecture Notes in Computer Science, Springer-Verlag, 176–184.

Aspvall, B., Plass, M. F. and Tarjan, R. (1979). A linear-time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters* **8** (3) 121–123.

https://doi.org/10.1017/S096012952000016X Published online by Cambridge University Press

Beame, P. (2004). Proof complexity. In: Rudich, S. and Wigderson, A. (eds.) *Computational Complexity Theory*, IAS/Park City mathematics series, vol. 10, American Mathematical Society, 199–246.
Beame, P. and Pitassi, T. (1996). Simplified and improved resolution lower bounds. In: *37th Annual Symposium on Foundations of Computer Science*, Burlington, Vermont, 14–16 October 1996, IEEE, 274–282.
Beame, P. and Pitassi, T. (1997). Resolution and the weak pigeonhole principle. In: *CSL: 11th Workshop on Computer Science Logic*, LNCS, Springer-Verlag.
Beame, P. and Pitassi, T. (1998). Propositional proof complexity: Past, present, future. *Bulletin of the EATCS* **65** 66–89.
Buss, S. R. (1999). Propositional Proof Complexity: An Introduction. In: Berger, U. and Schwichtenberg, H. (eds.) *Computational Logic*, Berlin, Springer-Verlag, 127–178.
Cormen, T. H., Leiserson, C. E., Rivest, R. L. and Stein, C. (2009). *Introduction to Algorithms*, 3rd edn., Cambridge, MA, The MIT Press.
Cook, S. A. and Reckhow, R. A. (1974). On the lengths of proofs in the propositional calculus (preliminary version). In: *Proceedings of the 6th Annual ACM Symposium on Theory of Computing, 30 April–2 May, 1974, Seattle, Washington, USA*, 135–148.
Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*, San Francisco, W. H. Freeman Company.
Haken, A. (1985). The intractability of resolution. *Theoretical Computer Science* **39** (2–3) 297–308.
Harrison, J. (2009). *Handbook of Practical Logic and Automated Reasoning*, 1st edn., Cambridge University Press.
Iwama, K. and Miyano, E. (1995). Intractability of read-once resolution. In: *Proceedings of the 10th Annual Conference on Structure in Complexity Theory (SCTC'95)*, Los Alamitos, CA, USA, June 1995, IEEE Computer Society Press, 29–36.
Iwama, K. (1997). Complexity of finding short resolution proofs. *Lecture Notes in Computer Science* **1295** 309–319.
Kleine Büning, H., Wojciechowski, P. J. and Subramani, K. (2017). The complexity of finding read-once nae-resolution refutations. In: *Logic and Its Applications - 7th Indian Conference, ICLA 2017, Kanpur, India, 5–7 January, 2017, Proceedings*, 64–76.
Kleine Büning, H., Wojciechowski, P. J. and Subramani, K. (2018). Finding read-once resolution refutations in systems of 2CNF clauses. *Theoretical Computer Science* **729** 42–56.
Kleine Büning, H. and Zhao, X. (2002). The complexity of read-once resolution. *Annals of Mathematics and Artificial Intelligence* **36** (4) 419–435.
Moore, C. and Mertens, S. (2011). *The Nature of Computation*, 1st edn., Oxford University Press.
Papadimitriou, C. H. (1994). *Computational Complexity*, New York, Addison-Wesley.
Porschen, S., Schmidt, T., Speckenmeyer, E. and Wotzlaw, A. (2014). XSAT and NAE-SAT of linear CNF classes. *Discrete Applied Mathematics* **167** 1–14.
Robinson, J. A. (1965). A machine-oriented logic based on the resolution principle. *Journal of the ACM* **12** (1) 23–41.
Rödl, V. and Siggers, M. H. (2006). Color critical hypergraphs with many edges. *Journal of Graph Theory* **53** (1) 56–74.
Schaefer, T. J. (1978). The complexity of satisfiability problems. In: Aho, A. (eds.) *Proceedings of the 10th Annual ACM Symposium on Theory of Computing*, New York City, NY, ACM Press, 216–226.
Subramani, K. and Gu, X. (2011). Absorbing random walks and the NAE2SAT problem. *International Journal of Computer Mathematics* **88** (3) 452–467.
Szeider, S. (2001). NP-completeness of refutability by literal-once resolution. In: *Automated Reasoning, First International Joint Conference, IJCAR 2001, Siena, Italy, 18–23 June, 2001, Proceedings*, 168–181.
Tseitin G.S. (1983). On the Complexity of Derivation in Propositional Calculus. In: Siekmann J.H., Wrightson G. (eds.) *Automation of Reasoning. Symbolic Computation (Artificial Intelligence)*, Berlin, Heidelberg, Springer, 466–483.
Urquhart, A. (1995). The complexity of propositional proofs. *The Bulletin of Symbolic Logic* **1** (4) 425–467.

Cite this article: Kleine Büning H, Wojciechowski P and Subramani K (2020). NAE-resolution: A new resolution refutation technique to prove not-all-equal unsatisfiability. *Mathematical Structures in Computer Science* **30**, 736–751. https://doi.org/10.1017/S096012952000016X