The complexity of Boolean formula minimization

David Buchfuhrer[*]
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
dave@cs.caltech.edu

Christopher Umans[†]
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
umans@cs.caltech.edu

Abstract

The Minimum Equivalent Expression problem is a natural optimization problem in the second level of the Polynomial-Time Hierarchy. It has long been conjectured to be Σ_2^P-complete and indeed appears as an open problem in Garey and Johnson [GJ79]. The depth-2 variant was only shown to be Σ_2^P-complete in 1998 [Uma98, Uma01], and even resolving the complexity of the depth-3 version has been mentioned as a challenging open problem. We prove that the depth-k version is Σ_2^P-complete under Turing reductions for all $k \geq 3$. We also settle the complexity of the original, unbounded depth Minimum Equivalent Expression problem, by showing that it too is Σ_2^P-complete under Turing reductions.

[*]Supported by NSF CCF-0830787, and BSF 2004329.
[†]Supported by NSF CCF-0830787 and a Sloan Research Fellowship.

1 Introduction

Circuit minimization problems are natural optimization problems contained in the second level of the Polynomial-Time Hierarchy (PH). The general form of such a problem is: given a Boolean circuit, find the smallest Boolean circuit that computes the same function. The input and output circuit may be required to be circuits of a particular form, e.g., Boolean formulas, or bounded-depth circuits. These problems are central problems in the field of logic synthesis, where fairly large instances are routinely solved using heuristics [DGK94]. They are also the prime examples of natural problems that should be complete for the classes of the second level of the PH. Indeed, versions of these problems inspired the definition of the PH in the early 70s by Meyer and Stockmeyer [MS72, Sto76], and Garey and Johnson use the formula variant to motivate the definition of the second level of the PH [GJ79]. See [SU02] for an up-to-date list of known complete problems in higher levels of the PH.

Completeness proofs for circuit minimization problems have been hard to find. The DNF formula version of circuit minimization was only proven to be Σ_2^P-complete in 1998 by Umans [Uma98, Uma01]; the other variants have remained prominent open problems. The only non-trivial hardness result for the general formula variant – called Minimum Equivalent Expression – is a $\mathrm{P}_{||}^{\mathrm{NP}}$-hardness result of Hemaspaandra and Wechsung from 1997 [HW97, HW02]. One reason reductions for these problems are difficult is that one direction of the reduction entails proving a lower bound for the type of circuit under consideration. This shouldn't be an absolute barrier, though, for two reasons. First, we have lower-bound proof techniques for Boolean formulas and bounded-depth circuits; nevertheless incorporating these into a reduction seems tricky. Second, a reduction need not entail *strong* lower bounds and in principle even slightly non-trivial lower bounds could suffice. A similar difficulty for potential reductions showing the (conjectured) NP-hardness of a related problem[1] was noted by Cai and Kabanets [KC00], although there, the use of weak lower bounds is not even an option, under a complexity assumption.

Proving Σ_2^P-completeness of the depth-3 formula variant was proposed [UVSV06] as a challenging first step, one that might begin to utilize techniques for proving lower bounds for bounded depth circuits (e.g., the Switching Lemma). In this paper we resolve, in one shot, the depth-3 case, as well as the depth-k variants for all $k \geq 3$. The same techniques show in addition that the unbounded depth Minimum Equivalent Expression problem is Σ_2^P-complete under Turing reductions. Our results resolve the complexity of these problems in the sense that they show for the first time that they are complete for the second level of the PH. Of course, many-one reductions would give a more refined result. We are able to achieve our results by exploiting the second way around the apparent barrier of proving circuit lower bounds: our reductions entail circuit lower bounds, but we get by with very weak ones, that with some effort are incorporated naturally into the structure of the reduction.

1.1 Description of the reduction

In this section we give a high-level description of the reduction, emphasizing a few interesting features before delving into the technical details.

The problem we reduce from is SUCCINCT SET COVER, which was defined and shown to be Σ_2^P-complete in [Uma99]:

Problem 1.1 (SUCCINCT SET COVER (SSC))**.** *Given a DNF formula D on variables*

$$v_1, \ldots, v_m, x_1, \ldots, x_n$$

[1]The problem is called "Circuit Minimization," but it and related problems in [AKRR03, AHM$^+$08, AKRR] refer to the NP problem of finding a minimum sized circuit which computes a given truth table.

and an integer k, is there a subset $I \subseteq \{1, 2, \ldots n\}$ with $|I| \leq k$ and $D \vee \bigvee_{i \in I} \overline{x_i} \equiv true$?

This can be seen as a succinct version of SET COVER, in which the $n+1$ exponentially large sets are implicitly and succinctly specified by the formulas $D, \overline{x_1}, \overline{x_2}, \ldots, \overline{x_n}$, and D is mandatory in any set cover. Here, the universe is the set of assignments to the variables, $\{true, false\}^{m+n}$. An implicitly specified set contains the assignments that it accepts.

We will assume that the formula D accepts the all-true assignment, as it only requires polynomial time to check this and the SSC instance is trivially false otherwise.

Our reductions exploit the special structure of this SUCCINCT SET COVER instance. In particular, all of the sets other than the one implicitly specified by D have an extremely simple form (they are just halfspaces), and in our reduction to MINIMUM EQUIVALENT EXPRESSION the choice of whether they are included or excluded from a cover will manifest itself relatively easily in the size of a minimum equivalent expression. However, D may be a complicated function, one whose minimum formula size is not readily apparent. To circumvent this problem, we will use a Turing reduction (actually a non-adaptive, or *truth-table*, reduction) which first ascertains the minimum formula size of D, and then asks one further query on a formula that incorporates D and other components, to determine whether or not the original instance of SUCCINCT SET COVER is a positive instance. This provides a somewhat rare example of a natural problem for which a Turing reduction seems crucial (in the sense that we do not know of any simple modification or alternative methods that would give a many-one reduction).

More specifically, the main idea of our reduction is to consider the following formula, derived from an instance of SUCCINCT SET COVER:

$$D \vee [z \wedge (\overline{x_1} \vee \cdots \vee \overline{x_n})] \tag{1}$$

where z is a new variable. Notice that when z is *false*, this formula is equivalent to just D, which (intuitively) forces a minimum equivalent formula to devote part of its size to computing D exactly. When z is $true$, the formula covers exactly the union of all of the sets in the instance of SUCCINCT SET COVER. That problem asks whether the disjunction of k or fewer $\overline{x_i}$ literals suffice to accept everything not accepted by D. If the variables indexed by $I \subseteq \{1, 2, \ldots, n\}$ suffice, then a very economical equivalent formula to the one above is

$$D \vee \left[z \wedge \left(\bigvee_{i \in I} \overline{x_i} \right) \right].$$

By forcing a minimum equivalent formula to contain a copy of D, we can ensure that a smallest equivalent formula is indeed of this intended form. We can then determine whether or not there is a cover of size k by asking whether (1) has an equivalent formula of size at most k greater than the size of the minimum subformula equivalent to D together with the z variable.

To make this actually work requires some modifications. For example, because the sets in the original instance cover all points in the domain, $D \vee z$ is already a small equivalent formula which does not depend at all on whether or not the instance of SUCCINCT SET COVER was a positive instance. But, we can solve this problem by modifying D initially to not accept the assignment in which every variable is set to $true$.

A more general technique that we use in several places in the reductions is "weighting" some variables in order to control the form of candidate small equivalent expressions. This is accomplished by replacing a single variable y with a conjunction of new variables, $y_1 \wedge \cdots \wedge y_w$, where w is the desired weight. We show that after this replacement, a minimum equivalent expression must be at least as large as the "w-minimum" expression (in which the size of a formula is measured by the number of occurrences of variables other than y plus w times the number of occurrences of the variable y).

We use this technique, for example, to weight z so highly that there can be only one occurrence of it; this then forms the conceptual pivot from which we argue that the subtrees of the formula surrounding that occurrence of z must compute D, and separately, a disjunction of as many variables as there are sets in a minimum cover of the original SUCCINCT SET COVER instance.

1.2 Outline.

In Section 2 we define general notation, and the variants of the problems we will be considering. In Section 3 we give the reductions – first a reduction showing that we can demand that the top-gate be an OR gate (or an AND gate) in Subsection 3.1, and then the main reductions in Subsection 3.2. We conclude in Section 4 with some open problems.

2 Preliminaries

Given a Boolean formula F, we use $|F|$ to mean the size of the formula F, measured by the number of occurrences of variables in F. Formulas can include constants, but these are not counted toward the size. We use $\overline{F}$ for the negation of the formula F. Similarly, for a variable x, $\overline{x}$ is the negation of x.

Restrictions. Given a function $f : \{\mathit{true}, \mathit{false}\}^n \to \{\mathit{true}, \mathit{false}\}$ and a function $\rho : [n] \to \{\mathit{true}, \mathit{false}, \mathit{free}\}$, we define the *restriction* of f to ρ, f_ρ to be the function which fixes the ith input to $\rho(i)$ if $\rho(i)$ is not equal to *free*, and leaves it as an input otherwise. Similarly, if F is a formula for f, we define F_ρ to be the formula in which every instance of the ith input variable is replaced with $\rho(i)$ if $\rho(i) \neq \mathit{free}$, and is unchanged otherwise. Note that F_ρ is a formula for f_ρ.

Weighted formulae. If the variables x_i of some function f have associated weights $w(x_i)$, then the *w-weighted size* of a formula for f is the sum of the weights of the variables occurring at the leaves (in their multiplicity). The usual measure of formula size is the w-weighted size when $w(x_i) = 1$ for all x_i. Note that, as usual, size counts the number of literals at the leaves, and not the $(\vee, \wedge, \neg)$ gates.

Given a weight function w, we can take a formula F and create a formula F' which has minimum formula size that is at least the minimum w-weighted formula size of F. Formula F' is obtained by substituting $x_i^{(1)} \wedge x_i^{(2)} \wedge \cdots \wedge x_i^{(w(x_i))}$ for every occurrence of x_i in F. Note that by moving negations to the variable level, we are substituting $\overline{x_i^{(1)}} \vee \overline{x_i^{(2)}} \vee \cdots \vee \overline{x_i^{(w(x))}}$ for every occurrence of $\overline{x_i}$. We call F' the *w-expanded version* of F. The following lemma demonstrates the usefulness of this transformation:

Lemma 2.1. *Let F be a formula and w a weight function for F. Let F' be the w-expanded version of F. Then the minimum size of a formula equivalent to F' is at least the minimum w-weighted size of a formula equivalent to F.*

Proof. Consider a minimum formula $\widehat{F'}$ equivalent to F'. For each x_i, let $1 \leq j_i \leq w(x_i)$ be the integer for which $x_i^{(j_i)}$ occurs least among the x_i-leaves of $\widehat{F'}$. Consider the restriction ρ that for each i sets $x_i^{(j)}$ to true for $j \neq j_i$. By our choice of j_i, $|\widehat{F'}|$ is at least the w-weighted size of $\widehat{F}'_\rho$. But the formula $\widehat{F}'_\rho$ clearly is equivalent to F, so its w-weighted size is an upper bound on the minimum w-weighted size of a formula equivalent to F. $\square$

2.1 The problems

As mentioned in the introduction, we will reduce from the Σ_2^P-complete problem SUCCINCT SET COVER. It will be convenient to work with a slightly modified version in which the goal is for the succinctly specified sets to cover everything *except* the all true assignment.

Problem 2.1 (MODIFIED SUCCINCT SET COVER (MSSC))**.** *Given a DNF formula D on variables*

$$v_1, v_2, \ldots, v_m, x_1, x_2, \ldots, x_n$$

and an integer k, is there a subset $I \subseteq \{1, 2, \ldots n\}$ with $|I| \leq k$ and for which

$$D \vee \bigvee_{i \in I} \overline{x_i} \equiv \left(\bigvee_{i=1}^{m} \overline{v_i} \vee \bigvee_{i=1}^{n} \overline{x_i} \right)?$$

It's easy to see that this variant of SUCCINCT SET COVER is Σ_2^P-complete by reducing from SUCCINCT SET COVER:

Theorem 2.2. MSSC *is Σ_2^P-complete.*

Proof. We are given an instance of SSC: a DNF D on variables

$$v_1, v_2, \ldots, v_m, x_1, x_2, \ldots, x_n$$

and an integer k. We produce the instance

$$D' = D \wedge \left(\bigvee_{i=1}^{m} \overline{v_i} \vee \bigvee_{i=1}^{n} \overline{x_i} \right)$$

(multiplied out into DNF) paired with the same integer k. Check in polynomial time whether D accepts the all true assignment. If not, produce any negative instance of MSSC, as the SSC instance is negative.

Otherwise, if there exists $I \subseteq [n]$ of size at most k for which $D \vee \bigvee_{i \in I} \overline{x_i} \equiv 1$ then clearly $D' \vee \bigvee_{i \in I} \overline{x_i}$ accepts everything except the all true assignment, and vice-versa, as we have already checked that D accepts the all true assignment. □

Remark 1. *In both* SSC *and* MSSC*, the instances produced by the reduction have the property that taking $I = \{1, 2, \ldots, n\}$ is a feasible solution. This clearly holds for* MSSC *if it holds for* SSC*. To see that it holds for* SSC*, refer to the reduction in [Uma99].*

The central problem we are concerned with in this paper is:

Problem 2.2 (MINIMUM EQUIVALENT EXPRESSION (MEE))**.** *Given a Boolean $(\wedge, \vee, \neg)$-formula F and an integer k, is there an equivalent $(\wedge, \vee, \neg)$-formula of size at most k?*

We also consider the constant-depth versions. When discussing constant-depth formulas, as usual, we allow arbitrary fan-in AND and OR gates and we use the convention that all NOT gates occur at the variable level.

Problem 2.3 (MINIMUM EQUIVALENT DEPTH-d EXPRESSION (MEE_d))**.** *Given a depth-d Boolean formula F and an integer k, is there an equivalent depth-d formula of size at most k?*

While distributing the NOT gates to the variable level clearly does not affect formula size, it's not as clear that finding the minimum depth-d formula is equivalent to finding the minimum depth-d formula with an OR gate at the root. The latter variant, defined below, will be easier to work with.

Problem 2.4 (MINIMUM EQUIVALENT DEPTH-d EXPRESSION WITH A TOP OR GATE ($\mathrm{MEE}_d{-}\mathrm{OR}$)). *Given a Boolean formula F with a top OR gate and an integer k, is there an equivalent depth-d formula with a top OR gate, of size at most k?*

Containment of these problems in Σ_2^P is trivial and well-known. In Theorem 3.3 we reduce $\mathrm{MEE}_d{-}\mathrm{OR}$ to MEE_d, so that Σ_2^P-hardness for the latter follows from the Σ_2^P-hardness for the former. Using $\leq_m$ to refer to many-one reductions and $\leq_{tt}$ to refer to truth table Turing reductions, the sequence of reductions used in this paper to show hardness of MEE_d is

$$\mathrm{SSC} \leq_m \mathrm{MSSC} \leq_{tt} \mathrm{MEE}_d{-}\mathrm{OR} \leq_m \mathrm{MEE}_d.$$

See Section 3.3 for the sequence of reductions to show hardness of MEE.

3 Main results

In this section we prove:

Theorem 3.1. *For every $d \geq 3$, the problem MEE_d is Σ_2^P-complete under polynomial-time Turing reductions.*

Theorem 3.2. *The problem MEE is Σ_2^P-complete under polynomial-time Turing reductions.*

These two theorems are proved via the reductions in Sections 3.1, 3.2, and 3.3. The first allows us to restrict our attention to the $\mathrm{MEE}_d{-}\mathrm{OR}$ problem, rather than the general MEE_d problem. This restriction allows us to focus on the $d \geq 3$ case, as $d = 2$ is simply DNF minimization, which was shown to be Σ_2^P-complete in [Uma01].

3.1 Top OR gate vs. unrestricted top gate

Theorem 3.3. *For every $d \geq 2$, there is a polynomial-time reduction from $\mathrm{MEE}_d{-}\mathrm{OR}$ to MEE_d.*

Proof. Fix $d \geq 2$. If every depth-d formula F has a minimum equivalent depth-d formula F' with an OR gate at the root, then the two problems are equivalent, and the identity reduction suffices.

Otherwise there exists a formula F^* such that F^* has a smaller equivalent depth-d formula with an AND at the root than the smallest equivalent depth-d formula with an OR at the root. Equivalently, $\overline{F^*}$ has a smaller equivalent depth-d formula with an OR at the root than the smallest equivalent depth-d formula with an AND at the root. Let G be a minimum depth-d formula for $\overline{F^*}$.

Now, given a depth-d formula F with a top OR gate and an integer k (which we may assume to be less than $|F|$), we create $|F| + 1$ copies of G on disjoint variable sets (also disjoint from the variable set of F). Call these copies G_i. Our reduction produces the formula

$$F' = F \vee G_1 \vee \cdots \vee G_{|F|+1} \tag{2}$$

paired with the integer $k' = (|F| + 1)|G| + k$.

If F has an equivalent depth-d formula with an OR at the root, of size at most k, then it is clear that F' has an equivalent depth-d formula of size at most k'.

If F does not have an equivalent depth-d formula with an OR gate at the root, of size at most $k \geq 0$, then we note that it cannot be a constant function. We wish to show that in this case F' does not have an equivalent depth-d formula of size at most k'. Suppose for the purpose of contradiction that it did, and call the equivalent formula $\widehat{F}'$. We claim that $\widehat{F}'$ must have an OR gate at the root. Note that G cannot be a constant function. Therefore, for each i there is a restriction ρ_i that sets the variables of F so that F evaluates to false, and sets the variables of G_j for $j \neq i$ so that G_j evaluates to false while leaving the variables of G_i free. The resulting formula $\widehat{F}'_{\rho_i}$ is equivalent to G_i, and if $\widehat{F}'$ had an AND gate at the root, this would be a depth-d formula for G_i with an AND gate at the root, which must have size at least $|G| + 1$. This holds for each i, and the G_i are on disjoint variable sets, so the total size of $\widehat{F}'$ must be at least $(|F| + 1)(|G| + 1)$, which is greater than k' (since we assumed that $k \leq |F|$).

Thus, $\widehat{F}'$ has size at most k' and an OR gate at the root. Since the G_i and F are not constant functions, and they are all on disjoint sets of variables, we can apply the restriction argument above to conclude that $(|F| + 1)|G|$ leaves must be used to account for the various G_i, and then at most $k' - (|F| + 1)|G| = k$ are available to compute F. Thus there must be an equivalent formula for F with an OR gate at the root, of size at most k (and this formula can be obtained by restricting the variables belonging to the G_i in $\widehat{F}'$ so that each G_i becomes false). So we conclude that F has an equivalent depth-d formula with an OR gate at the root of size at most k, a contradiction. □

3.2 Main reduction

The following is a Turing reduction from MSSC to $\mathrm{MEE}_d-\mathrm{OR}$. We describe the steps of a Turing Machine with access to an oracle for $\mathrm{MEE}_d-\mathrm{OR}$:

- We are given an instance of MSSC consistent with Remark 1: a DNF D on variables

$$v_1, v_2, \ldots, v_m, x_1, x_2, \ldots, x_n$$

  and an integer k. Let w be the weight function with $w(x_i) = 1$ for all i and $w(v_i) = n + 1$ for all i, and let D' be the w-expanded version of D. Note that D' has depth at most 3, as we are expanding a DNF formula.

- We make $O(\log |D'|)$ calls to the oracle to find the size u of the smallest equivalent depth-d formula with top OR gate for D', using binary search.

- Define the formula E involving fresh variables y_i and z as follows:

$$E = D \vee [(\overline{x_1} \vee \overline{x_2} \vee \cdots \vee \overline{x_n} \vee \overline{y_1} \vee \cdots \vee \overline{y_{u+n}}) \wedge z].$$

  Let w' be the weighting function with $w'(x_i) = 1$ for all i, $w'(v_i) = n + 1$ for all i, $w'(y_i) = 1$ for all i and $w'(z) = 2u + k + n + 1$, and let F be the w'-expanded version of E. We will label the copies of z used in the expanded version $z_1, z_2, \ldots, z_{2u+k+n+1}$. Note that F has only depth 3.

- We ask the oracle if F has an equivalent depth-d formula with top OR gate, of size at most $4u + 2k + 2n + 1$. We will show that the answer is "yes" iff the original MSSC instance was a positive instance.

Remark 2. *Note that since this reduction utilizes logarithmically many adaptive oracle calls, it can be transformed using standard techniques (see, e.g. [Pap94, Thm 17.7]) into a non-adaptive truth table Turing reduction utilizing polynomially many oracle calls. This is a nonadaptive AC^0 (or FO) reduction, so our main results could also be stated as $\Sigma_2^P \subseteq FO(\mathrm{MEE}), FO(\mathrm{MEE}_d)$.*

The remainder of this section is devoted to proving the following theorem:

Theorem 3.4. *Let $\widehat{F}$ be a minimum equivalent depth-d formula with top OR gate for F. Then $|\widehat{F}| \leq 4u + 2k + 2n + 1$ iff there exists $I \subseteq \{1, 2, \ldots, n\}$ with $|I| \leq k$ and and for which*

$$D \vee \bigvee_{i \in I} \overline{x_i} \equiv \left(\bigvee_{i=1}^{m} \overline{v_i} \vee \bigvee_{i=1}^{n} \overline{x_i} \right).$$

As a point of reference, Figure 1 shows the "intended" form of a minimum equivalent depth-d formula for F. Of course for one direction of the reduction we will need to show that a small formula must have this form, which is a somewhat involved argument.

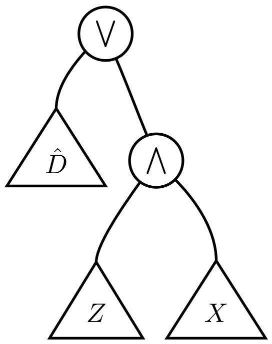

Figure 1: The desired form of an equivalent formula for F. Here $\widehat{D}$ is a minimum depth-d formula with top OR gate equivalent to D', $Z = \bigwedge_{i=1}^{2u+k+n+1} z_i$, and X is of the form $\bigvee_{i \in I} \overline{x_i} \vee \bigvee_{i=1}^{u+n} \overline{y_i}$.

In the backward (easy) direction, we claim that if the instance of MSSC is a positive instance, then there is a depth-d formula equivalent to F, of the form pictured in Figure 1, and with size at most $4u+2k+2n+1$. Let $\widehat{D}$ be a depth-d formula with top OR gate equivalent to D' of size u, and let $I \subseteq \{1, 2, \ldots, n\}$ be a set of size at most k for which

$$D \vee \bigvee_{i \in I} \overline{x_i} \equiv \left(\bigvee_{i=1}^{m} \overline{v_i} \vee \bigvee_{i=1}^{n} \overline{x_i} \right),$$

(such a set I exists because the MSSC instance is a positive instance). Then

$$\widehat{D} \vee \left[\left(\bigwedge_{i=1}^{2u+k+n+1} z_i \right) \wedge \left(\bigvee_{i \in I} \overline{x_i} \vee \bigvee_{i=1}^{u+n} \overline{y_i} \right) \right]$$

is a depth-d formula equivalent to F of size $4u + 2k + 2n + 1$. Furthermore, it is of the form pictured in Figure 1.

In the other direction, we assume that the MSSC instance is a negative instance, and we wish to show that there is *no* depth-d formula with top OR gate equivalent to F of size at most $4u + 2k + 2n + 1$. Let $\widehat{F}$ be a minimum depth-d formula for F.

We will derive from $\widehat{F}$ a minimum depth-d formula for F that is of the form pictured in Figure 1. We prove this in the next three subsections. Note that if $\widehat{F}$ has size larger than $4u + 2k + 2n + 1$, then we are done; therefore we will assume the contrary in what follows.

3.2.1 The z variable.

First, we show that there is some i for which z_i occurs exactly once in $\widehat{F}$ and that it does not occur negated.

Lemma 3.5. *If a formula for F has size at most $4u + 2k + 2n + 1$, then there is some i such that z_i occurs exactly once.*

Proof. If the formula is of size at most $4u + 2k + 2n + 1$ and yet contains two or more copies of each z_i, then this is a contradiction as the number of occurrences of z_i variables alone is $2(2u + k + n + 1) = 4u + 2k + 2n + 2 > 4u + 2k + 2n + 1$.

Thus, some z_i must occur at most once. Now, since we are assuming that D came from a negative instance of MSSC, we know that D rejects some assignment to its variables other than the all-true assignment. On the other hand E accepts this assignment when the z variable is true. This implies that E depends on z and that F (the w'-expanded version of E) depends on each z_i. So some z_i occurs *exactly* once. $\square$

Fix an i for which z_i occurs exactly once. Now, let ρ be the restriction that restricts all z_j for $j \neq i$ to $true$, and leaves all other variables free. From now on we will be working with $\widehat{F}_\rho$, which has only a single z variable.

Lemma 3.6. *Let f be the function corresponding to $\widehat{F}_\rho$. Further, let ρ_0 restrict z_i to false, leaving all other variables free and ρ_1 restrict z_i to $true$, leaving all other variables free. If z_i appears negated in $\widehat{F}_\rho$, then $f_{\rho_1} \Rightarrow f_{\rho_0}$.*

Proof. Consider any restriction σ that assigns all variables except z_i to true or false and leaves z_i free. $(\widehat{F}_\rho)_\sigma$ takes in the single input $\overline{z_i}$. Since there are no negations other than at the variable level, $(\widehat{F}_\rho)_\sigma$ is monotone in $\overline{z_i}$. Thus, $f_\sigma(true) \Rightarrow f_\sigma(false)$. Since this is true for all σ, $f_{\rho_1} \Rightarrow f_{\rho_0}$. $\square$

Now, if we substitute *false* for z_i in $\widehat{F}_\rho$, the function that this formula computes is equivalent to D', and if we substitute $true$, it accepts everything except the all true assignment to the x, y, and w-weighted v variables. These both follow directly from the construction of F and the fact that D does not accept the all true assignment. Defining f, ρ_0, ρ_1 as in Lemma 3.6 (and again using the fact that D comes from a negative instance of MSSC) we have that $f_{\rho_1} \not\Rightarrow f_{\rho_0}$. Lemma 3.6 then tells us that z_i occurs non-negated in the formula $\widehat{F}_\rho$.

3.2.2 Properties of $\widehat{F}_\rho$.

In the remainder of the proof, we will use ALLTRUE as shorthand for the all-true assignment to the variables of $\widehat{F}_\rho$ – namely, the x variables, y variables, the w'-expanded v variables, and z_i. Similarly $\overline{\text{ALLTRUE}}$ refers to the function that accepts every assignment to those variables except ALLTRUE. Here we define an important term used in the upcoming proofs.

Definition 3.1 (Apogee). *Let F be a depth-d formula with a top OR gate which contains exactly one copy of the sub-formula G. We call F a G-*apogee *if there is no other depth-d formula $F' \equiv F$ with a top OR gate, for which the following properties hold:*

- *F' contains exactly one copy of G.*
- *The one copy of G occurs at a higher level (closer to the top OR gate) in F' than in F.*
- *No variable occurs more often in F' than in F.*

Intuitively, F is a G-apogee if it is not possible to rearrange it so that G is higher in the formula.

Note that for any depth-d formula F with a top OR gate, and with one copy of sub-formula G, there exists a G-apogee F' equivalent to F with no variable occurring more often in F' than in F. Let $\widehat{F}'_\rho$ be a z_i-apogee equivalent to $\widehat{F}_\rho$ with size at most $|\widehat{F}_\rho|$. Note that $\widehat{F}'_\rho$ has exactly one copy of z_i, which is not negated. For future reference, we record a few other useful properties of $\widehat{F}'_\rho$:

Lemma 3.7. *The following properties regarding $\widehat{F}'_\rho$ hold:*

1. *$|\widehat{F}'_\rho| \leq |\widehat{F}| - (2u + k + n)$.*
2. *When z_i is true, $\widehat{F}'_\rho$ is equivalent to the formula $\overline{ALLTRUE}$ with z_i set to $true$.*
3. *When z_i is false, $\widehat{F}'_\rho$ is equivalent to D'.*

Proof. Property (1) follows from the observation in the proof of Lemma 3.5 that F depends on each z_j, so every z_j must appear at least once in $\widehat{F}$.

Property (2) follows because for formula E, when z is true, E accepts exactly those assignments with at least one variable false. This follows from the construction of E and Remark 1. This property is preserved when the v variables are w'-expanded. The resulting function is the same as the one obtained by w'-expanding z and then restricting via ρ, after replacing z with z_i.

Property (3) follows from the definition of F and ρ. □

3.2.3 The X subformula.

In this section we show (Lemma 3.9) that there is a minimum formula accepting *at least* all of the assignments to the v, x and y variables not accepted by D' and not accepting the all-true assignment of the form

$$\left(\bigvee_{i \in I} \overline{x_i} \vee \bigvee_{i=1}^{u+n} \overline{y_i} \right)$$

for some I. We will eventually use this to argue that z_i's sibling subformula in $\widehat{F}'_\rho$ has the intended form, and in a technical part of Section 3.2.4.

The following general lemma will be useful.

Lemma 3.8. *Let $\{t_1, t_2, \ldots, t_n\}$ be a set of variables, and S a subset of $\{true, false\}^n$. A smallest formula accepting at least the assignments in S but not accepting the all true assignment is of the form $\bigvee_{i \in I} \overline{t_i}$ for some $I \subseteq \{1, 2, \ldots, n\}$.*

Proof. Let T be a formula accepting at least S and rejecting the all-true assignment. Suppose that T depends on ℓ variables. Then $|T| \geq \ell$. Furthermore, in each assignment accepted by T, one of these variables is set to *false*, as T does not accept all-true. Therefore, if T depends on variables t_i for $i \in I$, the formula

$T' = \bigvee_{i \in I} \overline{t_i}$ accepts at least everything that T accepts. Furthermore T' does not accept all-true and $|T'| = \ell \leq |T|$.

Thus, given a minimum formula T that accepts at least S but not all-true, we can find another minimum formula accepting at least S but not all-true which is of the desired form. □

Applying the lemma in our setting yields:

Lemma 3.9. *Let S be the subset of assignments to the y variables plus the variables of D' (the w-expanded v variables and the x variables) which are not accepted by D' (ignoring the y variables). Then a minimum formula accepting at least S but not the all-true assignment to these variables is of the form $\bigvee_{i \in I} \overline{x_i} \vee \bigvee_{i=1}^{u+n} \overline{y_i}$ for some $I \subseteq \{1, 2, \ldots, n\}$.*

Proof. By Lemma 3.8, we know that a minimum formula for this will be a disjunction of negated variables (from among the x, y and w-expanded v variables). Note that for each i, S includes the assignment in which y_i is $false$, all the other y variables are true, and all the other variables are true, because D' does not accept the all-true assignment to its variables. Therefore, the disjunction must accept this assignment. On the other hand, flipping y_i to $true$ in this assignment results in the all-true assignment that the disjunction must not accept. Therefore the disjunction depends on each y_i, so it must contain all of the y variables.

Now, we simply need to see that none of the w-expanded v variables appear. If some $v_i^{(j)}$ does appear in the disjunction of negated variables, then it must be that D' rejects some assignment in which $v_i^{(j)}$ is $false$ and every other variable in the disjunction is true (otherwise $v_i^{(j)}$ could be safely omitted). But then by symmetry, D' also rejects some assignment in which $v_i^{(j')}$ is $false$ and every variable in the disjunction is true, for every $j' \neq j$ such that $v_i^{(j')}$ is not in the disjunction. Thus for all j', $v_i^{(j')}$ must be in the disjunction if $v_i^{(j)}$ is. So if a single v variable appears in the disjunction, then at least $n+1$ v variables appear (recall that $w(v_i) = n + 1$ for all i). However, by Remark 1, we know that the disjunction of the n negated x variables together with all of the negated y variables suffices. This is a smaller disjunction than any disjunction involving v variables, which would need to include $n + 1$ v variables (as argued above) together with all the y variables.

We conclude that a minimum formula accepting at least S but not the all-true assignment is of the claimed form. □

3.2.4 Position of the z variable.

Finally, we show (Lemma 3.13) that z_i occurs directly under a second-level AND gate in $\widehat{F}'_\rho$. We begin with two general lemmas

Lemma 3.10. *Let A be a subformula of formula G which has all negations pushed to the variable level, and suppose formula B implies G. Then, the formula obtained by replacing A with $A \vee B$ in G is equivalent to G.*

Proof. Because all of the negations have been pushed to the variable level, flipping the result of a non-input gate from $false$ to $true$ can only change the output of the formula from $false$ to $true$, and not the reverse. Since we are replacing A with $A \vee B$, this can only change the result of the top gate of A from $false$ to $true$. Furthermore, since B implies G, this can only occur when G is already true, and thus it will not change the output. □

Lemma 3.11. *Let A be a subformula of formula G that implies G. Then, the formula G' obtained by replacing A with false in G, and then taking the disjunction of this new formula with A is equivalent to G.*

Proof. In the case that A is true, then G must be true because A implies G. In this case G' will also be true, as A occurs directly beneath the top-level OR in G'. In the case that A is false, G' is equivalent to G with A replaced by $false$, so G' has the same result as G in this case as well. □

Note that the transformation in Lemma 3.11 does not increase the size of the formula, the number of occurrences of any variable, nor its depth if G already has a top OR gate. We now describe how the above general lemmas will be applied to $\widehat{F}'_\rho$:

Lemma 3.12. *Suppose that $\widehat{F}'_\rho$ has a subformula $A \wedge I$ as pictured in Figure 2. If $I \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$ and I does not have a top AND gate, then either $A \wedge I$ occurs at the second level or $\widehat{F}'_\rho$ is not an I-apogee.*

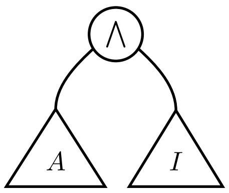

Figure 2: The subformula under consideration in Lemma 3.12

Proof. By assumption $I \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$, and so $A \wedge I \wedge \overline{\text{ALLTRUE}}$ does as well. If $A \wedge I \wedge \overline{\text{ALLTRUE}}$ is equivalent to $A \wedge I$, then $A \wedge I$ implies $\widehat{F}'_\rho$. Then by Lemma 3.11, we can move $A \wedge I$ to the second level without changing the formula or increasing the number of occurrences of any variable. So either $A \wedge I$ already occurred at the second level, or $\widehat{F}'_\rho$ is not an I-apogee.

Otherwise $A \wedge I$ accepts ALLTRUE. Since $A \wedge I$ accepts ALLTRUE, A must accept ALLTRUE, so $A \vee \overline{\text{ALLTRUE}}$ is true. By Lemma 3.10 and the assumption that $I \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$, we can replace $A \wedge I$ with

$$I \wedge (A \vee (I \wedge \overline{\text{ALLTRUE}})). \tag{3}$$

Note that if I is true, (3) reduces to $A \vee \overline{\text{ALLTRUE}}$, which is true, and if I is false, (3) is false. Thus, (3) is equivalent to I, so we can replace $A \wedge I$ with I, which places I higher in the formula (as I doesn't have a top AND gate, so the AND gate is not part of I and can be removed) without increasing the number of occurrences of any variable, demonstrating that $\widehat{F}'_\rho$ is not an I-apogee. □

Lemma 3.13. *z_i occurs directly under a second-level AND gate in $\widehat{F}'_\rho$.*

Proof. Recall that we have chosen $\widehat{F}'_\rho$ such that z_i occurs exactly once and non-negated. So we proceed by proving that z_i occurs under a second-level AND. The proof is by case analysis. There are four possible cases in which z_i does not occur under a second-level AND in $\widehat{F}'_\rho$: it can occur under the top-level OR gate, under an AND gate below the second level, under an OR gate below the third level, or under an OR gate at the third level. We will show that each of these cases results in a contradiction.

Case 1: The variable z_i cannot occur directly under the top OR gate, as setting z_i to $true$ would result in acceptance, and thus the formula would accept ALLTRUE, violating Lemma 3.7 (2).

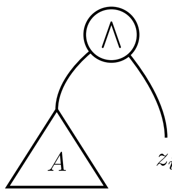

Figure 3: The portion of $\widehat{F}'_\rho$ containing z_i if z_i is under a low-level AND gate

Case 2: Suppose that z_i occurs under an AND gate below the second level. Then consider the subformula containing z_i, as pictured in Figure 3.

Since $z_i \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$ (by Lemma 3.7(2)) and $\widehat{F}'_\rho$ is a z_i-apogee, Lemma 3.12 (with I set to z_i) tells us that z_i is under a second-level AND gate, a contradiction.

Case 3: Suppose that z_i occurs under an OR gate below the third level. Then consider the subformula containing z_i, as pictured in Figure 4.

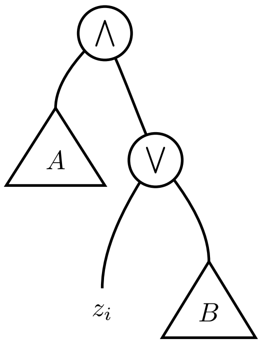

Figure 4: The portion of $\widehat{F}'_\rho$ containing z_i if z_i is under a low-level OR gate

We will show that $(z_i \vee B) \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$. We already know that $z_i \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$, so we only need to see that $B \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$. Now, $(z_i \wedge B) \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$ because $z_i \wedge \overline{\text{ALLTRUE}}$ does. And, since z_i only occurs once in $\widehat{F}'_\rho$, in disjunction with B, $(\overline{z_i} \wedge B) \wedge \overline{\text{ALLTRUE}}$ must also imply $\widehat{F}'_\rho$, as flipping z_i from $true$ to $false$ will not change the result of any gate in the formula if B is already true. This is because setting B to $true$ will satisfy the OR gate that z_i occurs under, and since z_i only occurs once it cannot affect the result of any other gate. Thus, $B \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$, as both $(z_i \wedge B) \wedge \overline{\text{ALLTRUE}}$ and $(\overline{z_i} \wedge B) \wedge \overline{\text{ALLTRUE}}$ do.

Thus, since $(z_i \vee B) \wedge \overline{\text{ALLTRUE}}$ implies $\widehat{F}'_\rho$, by Lemma 3.12, we know that either $\widehat{F}'_\rho$ is not a $(z_i \vee B)$-apogee or $A \wedge (z_i \vee B)$ occurs at the second level. $A \wedge (z_i \vee B)$ can't occur at the second level, as this would place z_i under a third-level OR gate, which is not the case we are examining. However, if $\widehat{F}'_\rho$ is not a $(z_i \vee B)$-apogee, then there exists a formula $\widehat{F}^*_\rho \equiv \widehat{F}'_\rho$ in which no variable occurs more often than in $\widehat{F}'_\rho$ and $z_i \vee B$ occurs at a higher level than in $\widehat{F}'_\rho$. This would also place z_i at a higher level, contradicting the fact that $\widehat{F}'_\rho$ is a z_i-apogee. This leaves us with Case 4 below.

Case 4: If z_i occurs directly under a third-level OR gate, then the formula has the form in Figure 5.

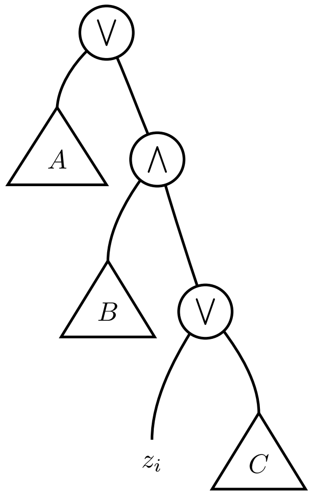

Figure 5: The form of $\widehat{F}'_\rho$ when z_i is under a 3rd-level OR gate

Case 4a: Suppose that C does not accept ALLTRUE. Then we claim that C implies $\widehat{F}'_\rho$. Consider an assignment accepted by $z_i \wedge C$. Since C does not accept ALLTRUE, such an assignment must have some variable false, and hence $\widehat{F}'_\rho$ accepts it (by Lemma 3.7 (2)). However, flipping z_i to false in this assignment cannot alter the output of C (since it does not contain z_i) and therefore the OR gate above C remains true, and no other gate values above it change, since z_i occurs only once in the formula. Thus $\widehat{F}'_\rho$ accepts this assignment as well. We conclude that $z_i \wedge C$ as well as $\overline{z_i} \wedge C$ imply $\widehat{F}'_\rho$, and therefore C implies $\widehat{F}'_\rho$.

Now, by Lemma 3.11, we can replace C with false and move C to the top-level OR gate. This leaves z_i alone under its OR gate, and so we can move it up one level so that it resides under the second level AND gate, contradicting that $\widehat{F}'_\rho$ is a z_i-apogee.

Case 4b: Suppose that C does accept ALLTRUE. Recall that D' does not accept ALLTRUE. We claim that C cannot accept any other assignment that D' does not accept. Suppose for the purpose of contradiction that it did, and let τ be an assignment to the variables of D' other than ALLTRUE that is accepted by C but not by D'. By Lemma 3.7 (3), when $z_i = \mathit{false}$, $\widehat{F}'_\rho$ does not accept τ, and by Lemma 3.7 (2), when $z_i = \mathit{true}$, $\widehat{F}'_\rho$ does accept τ (since τ is not the all true assignment). However, toggling z_i in this assignment cannot alter the output of $\widehat{F}'_\rho$, because z_i occurs only once, and under assignment τ, the subformula C already makes the OR above z_i true. This is a contradiction. Thus we know that $\overline{C}$ accepts at least everything not accepted by D', but not ALLTRUE, and then by Lemma 3.9, $\overline{C}$ has size at least $u + n$.

Referring again to Figure 5, we see that when restricting z_i to true in $\widehat{F}'_\rho$, the formula reduces to $A \vee B$. By Lemma 3.7 (2), the resulting formula accepts everything except ALLTRUE. Therefore $A \vee B$ depends on every variable other than z_i, and so every variable must appear at least once, and their combined size must be at least $u + n$ from the y variables alone.

Adding up the sizes of A, B, C and z_i, we have that $|\widehat{F}'_\rho|$ is at least $(u + n) + (u + n) + 1 = 2u + 2n + 1$. Applying Lemma 3.7 (1), we find that

$$|\widehat{F}| \geq (2u + 2n + 1) + (2u + k + n) = 4u + 3n + k + 1.$$

Since $k < n$ (the MSSC instance is trivially a positive instance if $k \geq n$) this quantity is strictly greater than $4u+2k+2n+1$, contradicting our original assumption that $|\hat{F}| \leq 4u+2k+2n+1$. We conclude that this sub-case cannot arise.

The only remaining case is that z_i already occurs under a second-level AND, completing the proof. $\square$

3.2.5 Finishing up.

Lemma 3.14. *There is a minimum depth-d formula with top OR gate equivalent to F, of the form pictured in Figure 6.*

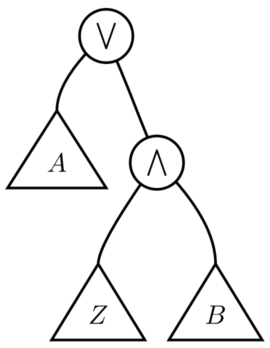

Figure 6: The required form of a minimum depth-d formula with top OR gate equivalent to F.

Proof. By Lemma 3.13, there is a minimum depth-d formula with top OR gate equivalent to $\widehat{F}_\rho$ that is of the form in Figure 6, but with the Z subformula replaced by z_i. Replacing z_i with $\bigwedge_{j=1}^{2u+k+n+1} z_j$, we obtain a depth-$d$ formula for F, of the form pictured in Figure 6, of size at most $|\widehat{F}_\rho| + 2u + k + n$. By Lemma 3.7 (1), this quantity is a lower bound on the size of a minimum depth-d formula with top OR gate equivalent to F, so it must be minimum. $\square$

Now we are finally able to argue that $|\widehat{F}|$ must be larger than $4u + 2k + 2n + 1$. First, observe that by Lemma 3.14, there is a depth-d formula equivalent to F of the form pictured in Figure 6 whose size is the same as the size of $\widehat{F}$. In this formula, when Z is set to false, the function simplifies to just A, and by the definition of F, this must be equivalent to D', and hence it must have size at least u. On the other hand, when Z is set to $true$, the formula must accept every assignment in which at least one variable is set to $false$. This means that B must accept everything not accepted by D' except the all true assignment. By Lemma 3.9, we know that we can assume B to be a disjunction of negated variables, and that it must have size at least $u + n + k + 1$ (because the original MSSC instance was a negative instance). Adding the sizes of A and B to the number of z_i variables in Z, we have a formula of size at least $4u + 2k + 2n + 2$. We conclude that

$$|\widehat{F}| \geq 4u + 2k + 2n + 2 > 4u + 2k + 2n + 1$$

as required.

This completes the proof of Theorem 3.4.

3.3 The unbounded depth case

The reduction to MEE is the same as the reduction in the previous section (3.2). We never used the depth-d restriction in any of the arguments in that reduction; it was only mentioned in the context of ensuring that various manipulations *maintained* depth-d, a constraint that is no longer operative for the unbounded depth case. Furthermore, we can assume that an unbounded depth formula has a top OR gate, as a formula can be placed beneath an OR gate without an increase in size. So the sequence of reductions becomes

$$\mathrm{SSC} \le_m \mathrm{MSSC} \le_{tt} \mathrm{MEE}.$$

It is still convenient in the reduction to think of formulas with alternating levels of unbounded-fan-in AND and OR gates, and here we simply note that discussing the size of such formulas is the same as discussing the size of standard, fan-in-2 $(\wedge, \vee, \neg)$-formulas.

Proposition 3.15. *If F is a formula with unbounded-fan-in AND and OR gates of size s, then there is an equivalent formula F' with fan-in-2 AND and OR gates of size s. Similarly if there is a formula F with fan-in-2 AND and OR gates of size s, there is an equivalent formula F' with unbounded-fan-in AND and OR gates of size s.*

So hardness holds for unlimited depth formulas regardless of whether fan-in is bounded.

4 Conclusions and open problems

The most natural open problems remaining are to give many-one reductions for the problems in this paper (rather than Turing reductions), and to resolve the complexity of the circuit versions of the problems. Our techniques here rely heavily on the fact that we are dealing with formulas rather than circuits.

Another important direction is to study the approximability of the problems in this paper. For the depth-2 case (DNF minimization) it is known that the problems are inapproximable to within very large (N^{ε}) factors [Uma99]. Our reductions are quite fragile and do not seem to give any hardness of approximation results for these problems.

Finally, we note that the complexity of the Π_2^P versions of all of these problems remain open. These are problems of the form: given a Boolean circuit (of some specified form), is it a minimum circuit (of the specified form). Even for DNF formulas, this problem is not known to be Π_2^P-complete, although we conjecture that it is complete for that class.

Acknowledgements

We thank Edith Hemaspaandra for helpful comments on a draft of this paper, and the anonymous reviewers for their comments and corrections.

References

[AHM+08] E. Allender, L. Hellerstein, P. McCabe, T. Pitassi, and M. E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008.

[AKRR] E. Allender, M. Koucký, D. Ronneburger, and S. Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. To appear in JCSS.

[AKRR03] E. Allender, M. Koucký, D. Ronneburger, and S. Roy. Derandomization and distinguishing complexity. In *IEEE Conference on Computational Complexity*, pages 209–220. IEEE Computer Society, 2003.

[DGK94] S. Devadas, A. Ghosh, and K. Keutzer. *Logic synthesis*. McGraw-Hill, Inc., New York, NY, USA, 1994.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[HW97] E. Hemaspaandra and G. Wechsung. The minimization problem for Boolean formulas. In *FOCS*, pages 575–584, 1997.

[HW02] E. Hemaspaandra and G. Wechsung. The minimization problem for boolean formulas. *SIAM J. Comput.*, 31(6):1948–1958, 2002.

[KC00] V. Kabanets and J.-Y. Cai. Circuit minimization problem. In *STOC*, pages 73–79, 2000.

[MS72] A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *FOCS*, pages 125–129. IEEE, 1972.

[Pap94] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Sto76] L. J. Stockmeyer. The polynomial-time hierarchy. *Theor. Comput. Sci.*, 3(1):1–22, 1976.

[SU02] M. Schaefer and C. Umans. Completeness in the polynomial-time hierarchy: a compendium. *SIGACT News*, September 2002. Updated version available at `http://ovid.cs.depaul.edu/documents/phcom.pdf`.

[Uma98] C. Umans. The minimum equivalent DNF problem and shortest implicants. In *FOCS*, pages 556–563, 1998.

[Uma99] C. Umans. Hardness of approximating Σ_2^P minimization problems. In *FOCS*, pages 465–474, 1999.

[Uma01] C. Umans. The minimum equivalent DNF problem and shortest implicants. *J. Comput. Syst. Sci.*, 63(4):597–611, 2001.

[UVSV06] C. Umans, T. Villa, and A. L. Sangiovanni-Vincentelli. Complexity of two-level logic minimization. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 25(7):1230–1246, 2006.