The Minimum Equivalent DNF Problem and Shortest Implicants

Christopher Umans*
Computer Science Division
University of California, Berkeley
umans@cs.berkeley.edu

May 25, 2000

Abstract

We prove that the Minimum Equivalent DNF problem is Σ_2^p-complete, resolving a conjecture due to Stockmeyer. We also consider the complexity and approximability of a related optimization problem in the second level of the Polynomial Hierarchy, that of finding shortest implicants of a Boolean function. We show that when the input is given as a DNF, this problem is complete for a complexity class above coNP utilizing $O(\log^2 n)$-limited nondeterminism. When the input is given as a formula or circuit, the problem is Σ_2^p-complete, and Σ_2^p-hard to approximate within factors of $n^{1/2-\epsilon}$ and $n^{1-\epsilon}$, respectively.

Keywords: DNF minimization, logic synthesis, Polynomial Hierarchy, shortest implicant, limited non-determinism, computational complexity, hardness of approximation.

1 Introduction

Circuit minimization problems have been extensively studied since the 1950's as important practical problems in logic synthesis, and, since the early 1970's, as natural problems in the second level of the Polynomial Hierarchy. These problems are widely believed to be Σ_2^p-complete but hardness proofs have been elusive. Indeed, the complexity of certain circuit minimization problems is the subject of a longstanding conjecture due to Stockmeyer. In his well-known polynomial hierarchy paper [Sto76], he proposes one of the simplest circuit minimization problems, the MINIMUM EQUIVALENT DNF problem (MIN DNF), as a likely Σ_2^p-complete problem. In this paper, we prove that MIN DNF is Σ_2^p-complete.

Our proof involves a variant of a related problem, that of finding a shortest implicant of a Boolean function. The connection between MIN DNF and implicant

*Supported in part by NSF grant CCR-9626361 and an NSF Graduate Research Fellowship.

problems is not surprising; since the introduction of the now-classic Quine-McCluskey algorithm [Qui52, McC56], most heuristic and exact methods for DNF minimization have been based on explicitly or implicitly generating the prime implicants of the input function (see [Cou94] and the introduction of [Str92] for an overview). Our results show that the close connection between MIN DNF and finding implicants is the source of at least one aspect of its difficulty.

We hope that the connection between implicant finding and DNF minimization can be exploited further, possibly in exploring the approximability of DNF minimization and related problems. To this end, we investigate the complexity and approximability of shortest implicant problems, which, as basic optimization problems in the second level of the hierarchy, seem to be a natural starting point for this research. Indeed, a variant of the shortest implicant problem was used in recent inapproximability results for some Σ_2^p minimization problems [Uma99]. Shortest implicant problems also provide a natural way to turn QSAT$_2$ into an optimization problem.

For the circuit and formula variants of the shortest implicant problem, we show that the problems are Σ_2^p-complete, and Σ_2^p-hard to approximate to within $n^{1-\epsilon}$ and $n^{1/2-\epsilon}$ factors for all $\epsilon > 0$, respectively. For the DNF variant, we find, somewhat surprisingly, that it is complete for a much weaker class in the second level of the hierarchy utilizing $O(\log^2)$-limited non-determinism.

1.1 Previous work

The number of papers on exact algorithms and heuristics for both DNF minimization and implicant finding is too great to survey here. We direct the reader to [Cou94] and [DGK94] for an overview. Comparatively little research has focused on the computational complexity of these problems. Since its introduction in [Sto76], no significant progress was reported on the hardness of MIN DNF until the recent paper by Hemaspaandra and Wechsung [HW97], in which they prove that MIN DNF is $P_{||}^{NP}$-hard. Their proof relies on a result of Wagner [Wag87] that allows certain NP-hardness results to be boosted to $P_{||}^{NP}$-hardness results. (As noted in [HW97], a proof that MIN DNF is NP-hard was already implicit in [Gim65]). Our approach is different, and we are able to show Σ_2^p-completeness directly with a two-step reduction.

To our knowledge, this paper is the first to consider the computational complexity of the shortest implicant problem, although variants of the problem are similar to certain "circuit subfunction relation" problems shown to be Σ_2^p-complete in [BR93]. For a survey of a handful of other problems known to be complete for the second level of the hierarchy, see the introduction to [Sch98].

1.2 Practical significance

In the more common case of NP-complete problems, a completeness result removes the likelihood of an efficient exact solution and justifies a concentration of effort on approximate or heuristic solutions. It might be argued that for circuit minimization and shortest implicant problems, for which coNP-hardness can be trivially proved, more precise complexity characterizations are of little more than academic interest to practitioners. However, one quickly realizes that most reasonable methods for approximately or heuristically solving these problems requires the exact solution of TAUTOLOGY as a "subroutine." Indeed, at least one major software suite for logic minimization (ESPRESSO) utilizes just such a TAUTOLOGY subroutine [DGK94], which of course has been tuned to work reasonably quickly on most real-world instances. In this context an "efficient" algorithm is an algorithm in P^{NP}, and a hardness result requires placing the problem in a complexity class above NP or coNP. It is therefore appropriate to investigate the complexity of these problems *relative to an NP oracle*, which requires precisely the sort of characterizations in the second level of the hierarchy that we provide in this paper.

2 Preliminaries

Let f be a Boolean function. An *implicant* of f is a conjunction C of literals for which $C \Rightarrow f$. We say that a conjunction C *covers* a truth assignment to f if the assignment makes C true; an implicant of f covers only assignments that make f true. For each well-formed conjunction C (i.e., one that does not include both a variable and its negation), there is a natural associated partial assignment that sets the variable x_i to 1 if the literal x_i appears in C and to 0 if the literal $\overline{x_i}$ appears in C. It is clear that C is an implicant if and only if applying this partial assignment to f results in a tautology. By an abuse of notation, we will denote by f_C the function obtained by applying the partial assignment associated with conjunction C. Under this interpretation, we can think of an implicant of f as a partial assignment that forces f to output 1.

It will often be notationally convenient to treat conjunctions, and implicants in particular, as sets of *literals* (variables or their negations). That is, when we write $C_1 \subseteq C_2$, we mean that C_1 contains a subset of the literals in C_2. It should be noted that in the logic synthesis literature, conjunctions are often viewed as representing subcubes of the Boolean hypercube, and so containment is written in the opposite direction ($C_1 \supseteq C_2$, read as "C_1 subsumes C_2"). If we wish to refer to the set of variables occurring in C (as opposed to the literals), we write $\mathrm{vars}(C)$.

The *length* of a conjunction C is simply $|C|$, the number of literals in the conjunction. The size of a formula ϕ, denoted $|\phi|$, is the number of occurrences of literals in the formula. For convenience, we denote by $SI(f)$ the length of the shortest implicant of f.

A conjunction (assignment, partial assignment, etc.) C_1 is *inconsistent* with another conjunction (assignment, partial assignment, etc.) C_2 if one includes literal x and the other includes literal $\overline{x}$. Otherwise, they are *consistent*. Frequently, in proofs, we will need to demonstrate that a conjunction C is *not* an implicant of f. We do this by exhibiting a truth assignment A that is consistent with C, but for which $f(A) = 0$. Such an assignment is a *witness* that C is not an implicant of f.

2.1 Problems

The circuit minimization problem we are concerned with in this paper is the following:

> MINIMUM EQUIVALENT DNF (MIN DNF): Given a DNF formula ϕ and an integer k, is there a DNF formula equivalent to ϕ with k or fewer occurrences of literals?

As noted in the introduction, we will be led naturally to a number of problems having to do with finding shortest implicants. The general shortest implicant problem is defined as follows:

> SHORTEST IMPLICANT: Given a Boolean function ϕ and an integer k, is there an implicant of ϕ of length at most k?

This is not yet a complete description; we still need to specify a input representation for ϕ. We consider three variants of the shortest implicant problem: SHORTEST IMPLICANT[DNF], in which ϕ is given as a DNF formula; SHORTEST IMPLICANT[FORMULA], in which ϕ is given as a formula over $(\wedge, \vee, \neg)$; and SHORTEST IMPLICANT[CIRCUIT], in which ϕ is given as a circuit over $(\wedge, \vee, \neg)$.

For completeness, we define QSAT_2, the canonical Σ_2^p-complete problem [Sto76], from which most of the reductions begin:

> QSAT_2: Given disjoint sets of variables X_1 and X_2, and a 3-DNF formula ϕ that uses exactly the variables in $X_1 \cup X_2$, is the following statement true: $\exists X_1 \forall X_2\ \phi$?

Notice that this is equivalent to asking whether there exists an implicant C of ϕ such that $\text{vars}(C) \subseteq X_1$.

2.2 Tools

In this section we gather some common arguments used throughout the paper. First, we state some elementary properties of implicants:

Observation 1 (Properties of implicants) *Let f be a Boolean function.*

1. *If C_1 is an implicant of f and C_2 is a conjunction for which $C_1 \subseteq C_2$, then C_2 is an implicant of f.*
2. *If $f = f_1 \wedge f_2$, then every implicant of f must also be an implicant of f_1 and an implicant of f_2.*
3. *If $f = f_1 \vee f_2$, then every implicant of f_1 is an implicant of f.*

In the next lemma, we describe a transformation, *parity substitution*, that has the effect of "weighting" variables in shortest implicant problems. A similar technique is used in [BR93, BRS98].

Lemma 1 (Parity Substitution) *Let $f(x_1, x_2, \ldots x_n)$ be a Boolean function, and let $W \subseteq \{x_1, x_2, \ldots x_n\}$ be the set of variables to weight. Define Boolean function $f' = f(h_1, h_2, \ldots h_n)$ where h_i is an independent copy of the Boolean parity function on m variables if $x_i \in W$, and $h_i = x_i$ otherwise. Then the following hold:*

1. *$SI(f') = \min_C\{|vars(C) \setminus W| + m|vars(C) \cap W|\}$, where C ranges over all implicants of f.*

2. *if C is an implicant of f with $vars(C) \cap W = \emptyset$, then C is an implicant of f'.*

Proof: The second claim is obvious; the first requires proof. Consider a shortest implicant C' of f'. For each i, either h_i is *forced* by C' to zero or one, or else h_i is *free* – it may take on both values in assignments consistent with C'. For each h_i that is forced, C' must include an implicant of h_i or $\neg h_i$, which requires m literals if $x_i \in W$, and one literal otherwise. There is also a corresponding implicant C of f that sets x_i to zero or one for each i for which h_i is forced. For this C, we have that $|C'| \geq |\mathrm{vars}(C) \setminus W| + m|\mathrm{vars}(C) \cap W|$.

Similarly, for each implicant C of f, there exists a corresponding implicant C' of f' that sets h_i to zero or one for each variable x_i that appears in C. For each such x_i, if $x_i \in W$, this requires m literals to force the parity formula to zero or one; otherwise, it requires only a single literal (x_i or $\overline{x_i}$). This holds for all implicants C of f, so $SI(f') \leq \min_C\{|\mathrm{vars}(C) \setminus W| + m|\mathrm{vars}(C) \cap W|\}$. This completes the proof. ■

We now define a "choice" formula that will be used in two of the reductions in later sections of the paper. This construction will be used to force some structure on the implicants of a formula, by requiring that they "choose" one literal from each of n subsets:

Definition 1 ($(S_1, S_2, \ldots S_n)$-choice formula) *Given $n \geq 2$ disjoint sets of variables $S_1, S_2, \ldots S_n$, define an $(S_1, S_2, \ldots S_n)$-choice formula f as follows:*

$$f = \bigvee_{j=1}^{|S_1|} (s_{1j} z_1) \vee \bigvee_{i=2}^{n-1} \bigvee_{j=1}^{|S_i|} (\overline{z_{i-1}} s_{ij} z_i) \vee \bigvee_{j=1}^{|S_n|} (\overline{z_{n-1}} s_{nj}),$$

where $\{s_{i1}, s_{i2}, \ldots s_{i|S_i|}\}$ are the variables in S_i, and the z variables are new variables.

Lemma 2 *Let f be an $(S_1, S_2, \ldots S_n)$-choice formula, and let C be a conjunction consisting only of positively occurring variables from $\cup_i S_i$. Conjunction C is an implicant of f if and only if $|vars(C) \cap S_i| \geq 1$ for all i.*

Proof: ($\Rightarrow$) Suppose $\text{vars}(C) \cap S_i = \emptyset$ for some i. Then we can construct an assignment A consistent with C and for which $f(A) = 0$ as follows: A maps every variable not in C to zero, except z_k for all $k \geq i$, which it maps to one. Assignment A is a witness that C is not an implicant of f.

($\Leftarrow$) Given conjunction C of the type specified in the lemma, with $|\text{vars}(C) \cap S_i| \geq 1$ for all i, we can pare it down to obtain a conjunction C' such that $|\text{vars}(C') \cap S_i| = 1$ for all i. Conjunction C' is an implicant since it results from the consensus[1] of exactly n terms of f (specifically, the term that includes $(\text{vars}(C') \cap S_i)$, for each i). Since $C' \subseteq C$, and C' is an implicant of f, C is also an implicant of f. ■

We will use $(S_1, S_2, \ldots S_n)$-choice formulae in contexts where there is an upper bound of n on the length of an implicant. By the above lemma, such an implicant must include *exactly* one of the variables in each set S_i.

3 Complexity of DNF minimization

An obvious heuristic for reducing the length of a DNF is the following: for each term in the DNF, remove literals one at a time until the term is a minimal (or *prime*) implicant. This can clearly be done in polynomial time, provided an NP oracle is available to determine at each step whether the current term is still an implicant. Notice, however, that the length of the resulting DNF depends on the order that literals are removed from each term.

It is natural to ask whether this algorithm can be improved so that it shortens each term of the DNF *optimally*; that is, for each term t, the algorithm produces a *shortest* implicant t' among all implicants $t' \subseteq t$. The answer is likely "no" – we show in the next theorem that shortening a single term of a DNF optimally is Σ_2^p-complete; this provides a starting point for showing that MIN DNF is Σ_2^p-complete. The decision version of this problem is defined as:

> SHORTEST IMPLICANT CORE: Given a DNF formula $\phi = t_1 \vee t_2 \vee \ldots \vee t_n$, where the t_i are the terms of ϕ, and an integer k, does ϕ have an implicant $C \subseteq t_n$ of length at most k?

Theorem 3 SHORTEST IMPLICANT CORE *is* Σ_2^p*-complete.*

Proof: Let $\langle X_1, X_2, \phi \rangle$ be an instance of QSAT_2 and let $m = |X_1|$. We first construct a DNF ϕ' from ϕ as follows: introduce a set S of $2m$ new variables $a_1, a_2, \ldots a_m$ and $b_1, b_2, \ldots b_m$ and for each variable $x_i \in X_1$, replace occurrences of the literal x_i with a_i and occurrences of the literal $\overline{x_i}$ with b_i. Let S be the conjunction $a_1 b_1 a_2 b_2 \ldots a_m b_m$.

[1]The *consensus* of two conjunctions xA and $\overline{x}B$, where x is a variable and A and B are conjunctions of literals, is AB, unless A and B are inconsistent, in which case the consensus is said to not exist. If xA and $\overline{x}B$ are implicants of a function ϕ, then their consensus, if it exists, is also an implicant of ϕ.

Claim 1 *The instance of* QSAT$_2$ *is a positive instance if and only if there exists an implicant of ϕ' consisting of exactly one literal from every (a_i, b_i) pair.*

Proof: ($\Rightarrow$) If the instance of QSAT$_2$ is a positive instance, then there exists an implicant C of ϕ for which $\text{vars}(C) = X_1$. Define conjunction C' to contain a_i if the literal x_i appears in C and b_i if the literal $\overline{x_i}$ appears in C. Clearly $\phi'_{C'}$ includes every term in ϕ_C, and possibly additional terms. Therefore, since $\phi_C \equiv 1$, it must also be that $\phi'_{C'} \equiv 1$, which implies that C' is an implicant of ϕ'. And, C' consists of exactly one literal from every (a_i, b_i) pair, as required.

($\Leftarrow$) We prove the contrapositive. Assume the instance of QSAT$_2$ is a negative instance. This implies that for every conjunction C for which $\text{vars}(C) = X_1$, there is an assignment A consistent with C for which $\phi(A) = 0$. Hence, for every conjunction C' for which $\text{vars}(C') = \text{vars}(S)$ and that contains either $\{a_i, \overline{b_i}\}$ or $\{\overline{a_i}, b_i\}$ for every i, there is an assignment A' consistent with C' for which $\phi(A') = 0$. Every conjunction D consisting of exactly one literal from every (a_i, b_i) pair can be augmented (by adding literals) into a conjunction C' of the form just described, so for every such D, there exists a witness A' that it is not an implicant of ϕ'. ■

Let f be an $(\{a_1, b_1\}, \{a_2, b_2\}, \ldots \{a_m, b_m\})$-choice formula. Our final formula is

$$\phi'' = (f \wedge \phi') \vee S.$$

When the parenthesized expression is multiplied out into DNF, the length remains polynomial in the length of the original input formula ϕ.

Claim 2 *Formula ϕ'' has an implicant $C \subseteq S$ of length at most m if and only if the instance of* QSAT$_2$ *is a positive instance.*

Proof: ($\Rightarrow$) Suppose ϕ'' has an implicant $C \subseteq S$ of length at most m. It must contain at least one literal from every (a_i, b_i) pair, for if it does not, then by the proof of Lemma 2, there is an assignment *with some a_i or b_i set to zero* that witnesses the fact that C is not an implicant of f. For this assignment f is false and term S is false, so ϕ'' is false; hence C cannot be an implicant of ϕ'', a contradiction.

We now know that C contains at least one literal from every (a_i, b_i) pair, and since $|C| \leq m$, it must be that C contains *exactly* one literal from every (a_i, b_i) pair. Now suppose the instance of QSAT$_2$ is a negative instance. By the proof of Claim 1 above, there is an assignment with some a_i or b_i set to zero that witnesses the fact that C is not an implicant of ϕ'. For this assignment, ϕ' is false and term S is false; as above, this implies that C is not an implicant of ϕ'', a contradiction. Therefore, the instance of QSAT$_2$ is a positive instance.

($\Leftarrow$) By Claim 1, if the instance of QSAT$_2$ is a positive instance, then there exists an implicant of ϕ' consisting of exactly one literal from every (a_i, b_i) pair. By Lemma 2, a conjunction of this form is also an implicant of f, so it is an implicant of ϕ'', and it has length m as required. ■

The instance of SHORTEST IMPLICANT CORE given by $\langle\phi'', m\rangle$ is therefore a positive instance if and only if the instance of QSAT_2 is a positive instance. This completes the reduction. ■

Theorem 4 MIN DNF *is Σ_2^p-complete.*

Proof: We reduce SHORTEST IMPLICANT CORE to MIN DNF. Let $\langle\phi = t_1 \vee t_2 \vee \ldots \vee t_n, k\rangle$ be an instance of SHORTEST IMPLICANT CORE. The general idea of the reduction is to force any minimum length DNF that is equivalent to ϕ to include term t_i for all $i < n$. We accomplish this in two steps. First, we produce from ϕ another DNF formula $\phi' \vee t_n$ which is equivalent to ϕ and has the additional property that every assignment covered by t_n falsifies every term of ϕ'. Next, we modify this formula to produce a DNF formula ϕ'' for which (in essence) all terms except the last term, t_n, must be part of a minimum equivalent DNF. The size of the minimum equivalent DNF for ϕ'' is then directly related to the size of the shortest implicant core of the original instance.

To begin, let $\phi' = \phi \wedge \neg t_n$. When ϕ' is multiplied out into DNF, its length remains polynomial in the length of ϕ. Let $s_1, s_2, \ldots s_m$ be the terms of ϕ' when it is multiplied out into DNF, and let $m' = \max(m, k+1)$. We define the following formula:

$$\begin{aligned} \phi'' &= t_n w_1 w_2 \ldots w_{m'} \vee \bigvee_{i=1}^{m} s'_i, \\ & \quad \text{where } s'_i = s_i w_1 w_2 \ldots w_{i-1}\ w_{i+1} \ldots w_{m'}. \end{aligned}$$

Let $k' = k + m' + \sum_{i=1}^{m} |s'_i|$. Our intended instance of MIN DNF is $\langle\phi'', k'\rangle$. We begin with the easy half of the proof:

Claim 1 *If C is an implicant of ϕ for which $C \subseteq t_n$, then*

$$\phi'' \equiv C w_1 w_2 \ldots w_{m'} \vee \bigvee_{i=1}^{m} s'_i.$$

Proof: Removing literals from a term of ϕ'' does not alter the function computed, as long as the modified term is an implicant of ϕ''. Therefore it is sufficient to show that $C w_1 w_2 \ldots w_{m'}$ is an implicant of ϕ''. Notice that $\phi \equiv \phi' \vee t_n \equiv \phi''_B$, where $B = \{w_1, w_2, \ldots w_{m'}\}$. Since C is an implicant of ϕ, it is also an implicant of ϕ''_B, and therefore $C w_1 w_2 \ldots w_{m'}$ is an implicant of ϕ''. ■

If the implicant C in the previous claim has length $\leq k$, then the equivalent DNF given in the claim has length $\leq k'$; hence the instance of MIN DNF is a positive instance if the instance of SHORTEST IMPLICANT CORE is a positive instance.

Let σ be a DNF equivalent to ϕ'' of length at most k'. We show that the instance of SHORTEST IMPLICANT CORE must be a positive instance. We begin by showing that σ must include the term s'_i (in a loose sense), for all i.

Claim 2 *(1) For all i, σ includes a term s_i'' such that $s_i' \subseteq s_i''$, (2) the collective length of these terms is at least $\sum_{i=1}^{m} |s_i'|$, and (3) none of these terms covers any assignment covered by $t_n w_1 w_2 \ldots w_{m'}$.*

Proof: Let A_i be an assignment consistent with s_i' and with w_i mapped to 0, and let s_i'' be a term of σ that covers A_i. If s_i'' excludes any w_j for $j \neq i$, then s_i'' covers an assignment in which w_j and w_i are mapped to 0, but ϕ'' covers no such assignments. If s_i'' excludes any literal from s_i, then s_i'' covers an assignment inconsistent with s_i', but with w_i mapped to 0. This is a contradiction, since the only assignments with w_i mapped to 0 that make ϕ'' true are consistent with s_i'. So $s_i' \subseteq s_i''$, which implies that $|s_i'| \leq |s_i''|$. Since s_i'' contains the literal w_j for all $j \neq i$, it cannot cover A_j for $j \neq i$. Since the A_i are all distinct, the s_i'' are distinct, so their collective length is at least $\sum_{i=1}^{m} |s_i''| \geq \sum_{i=1}^{m} |s_i'|$. Finally, no s_i'' covers any assignment covered by $t_n w_1 w_2 \ldots w_{m'}$, since $s_i' \subseteq s_i''$ and our initial modification to ϕ ensures that s_i' and $t_n w_1 w_2 \ldots w_{m'}$ are disjoint. ■

In addition to the s_i'' terms, σ must include terms that collectively cover all the assignments covered by $t_n w_1 w_2 \ldots w_{m'}$. Since $|\sigma| \leq k'$, these additional terms must have total length at most $k' - \sum_{i=1}^{m} |s_i'| \leq k + m'$. Any implicant of ϕ'' that covers an assignment covered by $t_n w_1 w_2 \ldots w_{m'}$ must include the literals $w_1 w_2 \ldots w_{m'}$; otherwise it would cover an assignment consistent with t_n but with some w_i set to false, and ϕ'' covers no such assignments (recall our initial modification to ϕ). Therefore, each of the additional terms has length at least m', and since $k + m' < 2m'$, there must be only one such additional term. This term, C', has length at most $k + m'$, and $C' \subseteq t_n w_1 w_2 \ldots w_{m'}$. As noted above, $\phi_B'' \equiv \phi$, where $B = \{w_1, w_2, \cdots, w_{m'}\}$. Hence, letting $C = C' \setminus B$, we have $1 \equiv \phi_{C'}'' \equiv (\phi_C'')_B \equiv (\phi_B'')_C \equiv \phi_C$, so C is an implicant of ϕ, with $C \subseteq t_n$ and $|C| \leq k$; in other words, the instance of SHORTEST IMPLICANT CORE is a positive instance. This completes the reduction from SHORTEST IMPLICANT CORE to MIN DNF. ■

Note that this proof also shows that the following "easier" DNF minimization problem, which requires only "term-wise" minimality, is Σ_2^p-complete:

> TERM-WISE MIN DNF: Given a DNF formula $\phi = t_1 \vee t_2 \vee \ldots \vee t_n$, and an integer k, is there a DNF formula $\phi' = t_1' \vee t_2' \vee \ldots \vee t_n'$ equivalent to ϕ with k or fewer occurrences of literals, *and with the property that* $\forall i \;\; t_i' \subseteq t_i$?

4 Shortest implicant problems for formulae and circuits

The SHORTEST IMPLICANT CORE problem is a restricted version of a more natural optimization problem which simply asks for the shortest implicant of a Boolean function. In this section we give a simple proof that this problem is Σ_2^p-complete when the input is given as a formula or circuit. We also use parity substitution to establish that SHORTEST IMPLICANT[FORMULA] and SHORTEST IMPLICANT[CIRCUIT] are Σ_2^p-hard to approximate to within $n^{1/2-\epsilon}$ and $n^{1-\epsilon}$ factors, respectively.

Theorem 5 SHORTEST IMPLICANT[FORMULA] *and* SHORTEST IMPLICANT[CIRCUIT] *are* Σ_2^p*-complete.*

Proof: We give the reduction for SHORTEST IMPLICANT[FORMULA], which is a special case of SHORTEST IMPLICANT[CIRCUIT].

Let $\langle X_1, X_2, \phi \rangle$ be an instance of QSAT$_2$, and let $X_1 = \{x_1, x_2, \ldots x_m\}$. As noted previously, the instance is a positive instance if and only if there exists an implicant of ϕ consisting of only variables in X_1. We modify ϕ so that *all* implicants must include the variables in X_1 and m additional variables. In the resulting formula, an implicant consisting of *exactly* these $2m$ variables exists if and only if there is an implicant of ϕ consisting of only variables in X_1.

We introduce new variables $w_1, w_2 \ldots w_m$ and define ϕ' as follows:

$$\begin{aligned} \phi' &= \phi \wedge \bigwedge_{i=1}^{m} (x_i \oplus w_i) \\ &= \phi \wedge \bigwedge_{i=1}^{m} (x_i\overline{w_i} \vee \overline{x_i}w_i) \end{aligned}$$

We claim that ϕ' has an implicant of length at most $2m$ if and only if the instance of QSAT$_2$ is a positive instance.

($\Rightarrow$) Let C' be an implicant of ϕ'. Clearly vars(C') contains $x_1, x_2, \ldots x_m$ and $w_1, w_2, \ldots w_m$; otherwise there is an assignment consistent with C' in which $(x_i \oplus w_i)$ is false for some i. Because $|C'| \leq 2m$, vars(C') equals *exactly* $X_1 \cup \{w_1, w_2, \ldots w_m\}$. Since ϕ' is the conjunction of ϕ with additional formulae, C' is also an implicant of ϕ, which implies that there exists an implicant of ϕ consisting of exactly the variables in X_1 (simply restrict C' to the variables that appear in ϕ). Hence the instance of QSAT$_2$ is a positive instance.

($\Leftarrow$) If the QSAT$_2$ instance is a positive instance, then there is an implicant C of ϕ consisting of exactly the variables in X_1. Construct conjunction C' by augmenting C with w_i literals as follows: if C includes literal x_i, add $\overline{w_i}$; if C includes literal $\overline{x_i}$, add literal w_i. Conjunction C' is an implicant of ϕ' of length $2m$ as required. ■

We now proceed to showing the hardness of approximating SHORTEST IMPLICANT[FORMULA] and SHORTEST IMPLICANT[CIRCUIT]. The following observation is critical: in the proof of Theorem 5, it happens that *all* implicants of ϕ' contain at least one variable from X_2 if and only if the instance is a negative instance. In the following proofs, we use parity substitution to "weight" the variables in X_2. The result is that all implicants (and hence, all shortest implicants) of a negative instance are long, while a positive instance has a short implicant that avoids the weighted variables.

We will also use the following facts about parity formulae and circuits over $(\wedge, \vee, \neg)$ (see [Weg87]): the parity of n variables can be computed by a circuit with $O(n)$ gates, and by a formula with size $O(n^2)$.

The proofs of the next two theorems use reductions from QSAT$_2$ to SHORTEST IMPLICANT that produce a gap of g; i.e., given QSAT$_2$ instance I, the reduction produces a SHORTEST IMPLICANT instance I' and a bound k such that $SI(I') \leq k$ if I is a positive instance, but $SI(I') > g \cdot k$ if I is a negative instance. A reduction of this type shows that SHORTEST IMPLICANT is Σ_2^p-hard to approximate within a factor g, since a g-approximate algorithm could distinguish between positive and negative instances of QSAT$_2$.

In the next theorem, the value n in the approximation factor is the size of the formula (number of leaves).

Theorem 6 *Approximating* SHORTEST IMPLICANT[FORMULA] *to within an* $n^{1/2-\epsilon}$ *factor is* Σ_2^p*-hard, for any* $\epsilon > 0$.

Proof: Recall from the proof of Theorem 5 that we transformed an instance of QSAT$_2$ given by $\langle X_1, X_2, \phi \rangle$ into the following formula:

$$\phi' = \phi \wedge \bigwedge_{i=1}^{m} \left(x_i \overline{w_i} \vee \overline{x_i} w_i \right),$$

where $x_1, x_2, \ldots x_m$ are the variables in X_1. Let $y_1, y_2, \ldots y_p$ be the variables in X_2, and let $N = |\phi'|$. Let ϕ'' be the "weighted" version of ϕ', obtained by substituting, for each y_i, a parity formula on N^c new variables. Here, c is a constant that will depend on ϵ. The size of each substituted parity formula is $O(N^{2c})$, so $|\phi''| \leq \alpha N^{2c+1}$, where α is the constant in the big-oh notation.

As observed above, if the instance of QSAT$_2$ is a positive instance, then there exists an implicant C of ϕ' of length $2m$, consisting of exactly the variables $x_1, x_2, \ldots x_m$ and $w_1, w_2, \ldots w_m$. By Lemma 1, C is also an implicant of ϕ''. However, if the instance of QSAT$_2$ is a negative instance, then every implicant of ϕ' includes at least one of the y variables. Again by Lemma 1, this implies that $SI(\phi'') \geq N^c$.

Using the fact that $2m \leq N$, we have the following inequality for sufficiently large N and $c \geq (3/4)\epsilon^{-1}$:

$$\frac{N^c}{2m} > \left(\alpha N^{(2c+1)} \right)^{(1/2-\epsilon)},$$

which establishes the required gap. ■

Using the same method, we obtain an even stronger result for the version of the problem in which the input is a circuit. Here, the value n in the approximation factor is the size of the circuit (number of gates).

Theorem 7 *Approximating* SHORTEST IMPLICANT[CIRCUIT] *to within an* $n^{1-\epsilon}$ *factor is* Σ_2^p*-hard, for any* $\epsilon > 0$.

Proof: Analogous to the previous proof. We start with the same formula ϕ' but obtain *circuit* ϕ'' by substituting parity *circuits* on N^c variables for each y_i. The size of ϕ'' is at most αN^{c+1}. The same analysis as used above regarding the sizes of shortest implicants in the positive and negative cases holds.

The following inequality holds for sufficiently large N, and $c \geq 2\epsilon^{-1}$:

$$\frac{N^c}{2m} > \left(\alpha N^{(c+1)}\right)^{(1-\epsilon)}.$$

This establishes the required gap. ■

5 The shortest implicant problem for DNF formulae

For the special case of the shortest implicant problem in which the formula is required to be a DNF, the reduction used in the proof of Theorem 5 for formulae fails (it produces a function whose DNF representation is exponential in the worst case). Somewhat surprisingly, SHORTEST IMPLICANT[DNF] appears to be a much easier problem, due to the following combinatorial fact:

Lemma 8 *Let $\phi = t_1 \vee t_2 \vee \ldots \vee t_n$ be a DNF formula. Then any shortest implicant C of ϕ may be obtained from some term t_i with at most $\log n$ additions of literals and at most $\log n$ deletions of literals.*

Proof: Let C be a shortest implicant of ϕ, and let us rearrange the terms of ϕ so that $t_1, t_2, \ldots, t_{n'}$ are exactly those terms of ϕ that are consistent with C. Now, for all i, define $t'_i = t_i \wedge C$; this ensures that $C \subseteq t'_i$ for all i. We first argue that there must exist an i for which $|t'_i| \leq |C| + \log n'$. Suppose there were not. Then each t'_i has length greater than $|C| + \log n'$, and the maximum number of assignments covered by the terms $t'_1, t'_2, \ldots t'_{n'}$ is therefore strictly less than $n' 2^{m-|C|-\log n'} = 2^{m-|C|}$, where m is the number of variables in ϕ. This is a contradiction, since it implies that the terms consistent with C do not cover all of the assignments covered by C. So there must be an i for which $|t'_i| \leq |C| + \log n'$.

Conjunction t'_i was obtained from t_i by adding literals. We cannot have added more than $\log n'$ literals, unless $|t_i| < |C|$, which contradicts C being a shortest implicant of ϕ. Since $n' \leq n$, C is obtainable from t_i with at most $\log n$ insertions (to get t'_i) and at most $\log n$ deletions of literals (to recover C). ■

As a consequence of Lemma 8, SHORTEST IMPLICANT[DNF] requires only $O(\log^2)$-limited non-determinism and a coNP oracle. Given an instance $\langle \phi = t_1 \vee t_2 \vee \ldots \vee t_n, k \rangle$, we can guess an index i, a set I of $\log n$ literals to add, a set D of $\log n$ literals to delete, and construct a candidate conjunction $C = (t_1 \cup I) \setminus D$. We accept if C is well-formed, has length at most k, and a query to the coNP oracle indicates that it is an implicant of ϕ.

In fact, SHORTEST IMPLICANT[DNF] is *complete* for a complexity class encompassing all such problems that can be solved using $O(\log^2 n)$-limited nondeterminism and a coNP oracle. We call this class GC($\log^2 n$, coNP) after the "guess then check" framework of Cai and Chen [CC97] (however they are predominantly concerned with classes in this framework with far weaker verifiers). The class GC($\log^2 n$, coNP) can be described as all languages L expressible as:

$$L = \{I : \exists X \text{ such that } R(I, X)\},$$

where $|X| \leq O(\log^2 |I|)$ and R is a coNP predicate. Notice that the usual logical characterization of Σ_2^p is identical except that $|X|$ is bounded only by some polynomial in $|I|$.

It is straightforward to see that the class GC($\log^2 n$, coNP) lies between coNP and Σ_2^p in the polynomial hierarchy. It seems unlikely that GC($\log^2 n$, coNP) contains NP, although to our knowledge, formal evidence in support of this conjecture is not known. More on related complexity classes can be found in [CC97]. Limited non-determinism in the first level of the hierarchy has been studied by a number of authors [KF80, DT90, PY96].

Theorem 9 SHORTEST IMPLICANT[DNF] *is GC($\log^2 n$, coNP)-complete.*

Proof: Membership in the class follows from Lemma 8. Hardness is via a generic reduction. Let L be a language in GC($\log^2 n$, coNP), let R be the coNP predicate from the characterization of the class above, and let I be an instance. Let ϕ be a 3-DNF obtained from $R(I, X)$ via a Cook reduction. Formula ϕ consists of Boolean variables $x_1, x_2, \ldots x_k$ that constitute X (with $k = c \log^2 |I|$ for some constant c), and additional Boolean variables $y_1, y_2, \ldots y_m$. We assume that no term of ϕ contains more than one x literal; a formula lacking this property may be easily transformed into the desired form using standard methods (which, it should be noted, may introduce additional y variables, but not additional x variables). Instance I is in L if and only if there exists an implicant of ϕ consisting of exactly the variables $x_1, x_2, \ldots x_k$.

We think of the variables $x_1, x_2 \ldots x_k$ as being grouped into $r = \log |I|$ blocks of $s = k/r$ variables each. We group the terms of ϕ similarly, with ϕ_i containing the disjunction of exactly those terms of ϕ in which some variable in the i^{th} group appears (i.e., ϕ_1 contains those terms of ϕ that include one of $x_1, x_2, \ldots x_s$; ϕ_2 contains terms that include one of $x_{s+1}, x_{s+2}, \ldots x_{2s}$; etc...). Some terms in ϕ include no x variables at all, and we let ϕ_0 contain the disjunction of these terms. Because no term of ϕ contains more than one x variable, $\phi_0, \phi_1, \phi_2, \ldots, \phi_r$ partition the terms of ϕ, and of course $\phi \equiv \phi_0 \vee \phi_1 \vee \phi_2 \vee \cdots \vee \phi_r$.

There are $t = 2^s = |I|^c$ possible assignments to each group of x variables. Denote by ϕ_{ij} the formula ϕ_i in which the j^{th} assignment is applied to the x variables in the formula, for $1 \leq i \leq r$ and $1 \leq j \leq t$.

Let a_{ij} be new variables for $1 \leq i \leq r$ and $1 \leq j \leq t$, and let $A_i = \{a_{i1}, a_{i2}, \ldots a_{it}\}$ for each i. Define f to be an $(A_1, A_2, \ldots A_r)$-choice formula.

We are now one step away from the final formula. Define ϕ' as follows:

$$\phi' = f \wedge \left(\phi_0 \vee \bigvee_{i=1}^{r} \bigvee_{j=1}^{t} (a_{ij} \wedge \phi_{ij}) \right).$$

When multiplied out into DNF, ϕ' is a 7-DNF whose size is polynomial in $|I|$. Finally, define ϕ'' to be ϕ' in which all variables except for the a variables have been substituted for by parity on $r+1$ variables. Notice that the resulting formula is still only polynomially large when multiplied out into DNF, because the DNF representations of the parity formulae each have $2^{r+1}/2 = |I|$ terms, and there are at most a constant number of substitutions per term of ϕ'.

Claim 1 *Formula ϕ'' has an implicant of length at most r if and only if $I \in L$.*

($\Rightarrow$) If there exists an implicant of ϕ'' of length at most r, then there must exist an implicant C of ϕ' of length at most r that consists of only the a variables. If not, then $SI(\phi'') > r$ by Lemma 1, a contradiction. Implicant C must be an implicant of f and so by Lemma 2 it must include at least one $a_{ij} \in A_i$ for each $i \in \{1, 2, \ldots r\}$. Since $|C| \leq r$, it must include *exactly* one $a_{ij} \in A_i$ for each i. This implies that the following partial assignment A to exactly the x variables of ϕ makes ϕ_A a tautology: for each $i \in \{1 \ldots r\}$, the i^{th} group of x variables takes on the j^{th} possible value, where j is the unique index for which $a_{ij} \in A_i$ appears in C. To verify this fact, let C' be the conjunction created by augmenting C with $\overline{a_{ij}}$ for all a_{ij} not already in C. Since $C \subseteq C'$, C' is an implicant of ϕ'. By construction, ϕ_A is equivalent to $\phi'_{C'}$, which is a tautology; hence $\phi_A \equiv 1$, and therefore $I \in L$.

($\Leftarrow$) If $I \in L$, then there exists an implicant C of ϕ consisting of exactly the x variables. Define a conjunction C' as follows: for each $1 \leq i \leq r$, if C assigns the i^{th} group of x variables the j^{th} possible value, then include a_{ij} in C'. By Lemma 2, C' is an implicant of f, and C' is also an implicant of the parenthesized portion of ϕ' because its corresponding partial assignment preserves at least those terms in ϕ_A. Therefore C' is an implicant of ϕ', and by Lemma 1, C' is also an implicant of ϕ''. As required $|C'| \leq r$. ∎

6 Conclusions and Open Problems

By simply negating formulae, our results imply $\Sigma_2^{\rm p}$-completeness for CNF minimization. Similarly, the results for SHORTEST IMPLICANT[DNF] carry over to the SHORTEST IMPLICATE problem for a CNF formula (an *implicate* of a function f is an implicant of $\neg f$), and both the completeness and inapproximability results for SHORTEST IMPLICANT[FORMULA] and SHORTEST IMPLICANT[CIRCUIT] carry over to the problem of finding shortest implicates of formulae and circuits, respectively.

Very little has been proven about the complexity of the other major circuit minimization problems: the only non-trivial hardness result known for the Minimum Equivalent Expression problem is $\mathrm{P}_{||}^{\mathrm{NP}}$-hardness [HW97], and nothing better than trivial coNP-hardness is know for the Minimum Equivalent Circuit problem. The precise complexity is also unknown for many other variants of these problems, such as the "Π_2^p" versions ("Is C a minimal circuit?"). And, despite the continuing importance of these problems in logic synthesis and the significant effort spent designing heuristic and approximate solutions, few formal results (positive or negative) on the approximability of circuit minimization problems are known. The main remaining challenges are to settle the complexity and approximability of these problems.

We thank Christos Papadimitriou for suggesting this problem, and many useful discussions. We also thank Tiziano Villa for his careful reading of an earlier version of this paper, and the anonymous referees for several useful suggestions.

References

[BR93] Bernd Borchert and Desh Ranjan. The circuit subfunction relations are Σ_2^p-complete. Technical Report MPI-I-93-121, MPI Saarbrücken, 1993.

[BRS98] Bernd Borchert, Desh Ranjan, and Frank Stephan. On the computational complexity of some classical equivalence relations on Boolean functions. *Theory of Computing Systems*, 31(6):679–693, 1998.

[CC97] Liming Cai and Jianer Chen. On the amount of nondeterminism and the power of verifying. *SIAM Journal on Computing*, 26(3):733–750, 1997.

[Cou94] Olivier Coudert. Two-level logic minimization: an overview. *Integration, the VLSI journal*, 17(2):97–140, 1994.

[DGK94] Srinivas Devadas, Adhijit Ghosh, and Kurt Keutzer. *Logic Synthesis*. McGraw-Hill, Inc., 1994.

[DT90] Josep Diaz and Jacobo Toran. Classes of bounded nondeterminism. *Mathematical Systems Theory*, 23(1):21–32, 1990.

[Gim65] J. Gimpel. A method of producing a Boolean function having an arbitrarily prescribed prime implicant table. *IEEE Transactions on Computers*, 14:485–488, 1965.

[HW97] Edith Hemaspaandra and Gerd Wechsung. The minimization problem for Boolean formulas. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS '97)*, pages 575–84, 1997.

[KF80] C. M. R. Kintaal and P. C. Fischer. Refining nondeterminism in relativized polynomial-time bounded computations. *SIAM Journal on Computing*, 9(1):46–53, 1980.

[McC56] E. J. McCluskey. Minimization of Boolean functions. *Bell Systems Technical Journal*, 35(5):1417–44, 1956.

[PY96] Christos H. Papadimitriou and Mihalis Yannakakis. On limited nondeterminism and the complexity of the V–C dimension. *Journal of Computer and System Sciences*, 53(2):161–70, 1996.

[Qui52] W. V. Quine. The problem of simplifying truth functions. *American Mathematics Monthly*, 59(8):521–531, 1952.

[Sch98] Marcus Schaefer. Graph Ramsey theory and the polynomial hierarchy. Technical Report TR-98-04, University of Chicago, 1998. Extended abstract appeared in *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC '99)*.

[Sto76] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[Str92] Tadeusz Strzemecki. Polynomial-time algorithms for generation of prime implicants. *Journal of Complexity*, 8(1):37–63, 1992.

[Uma99] Christopher Umans. Hardness of approximating Σ_2^p minimization problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS '99)*, pages 556–563, 1999.

[Wag87] K. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoretical Computer Science*, 51(1–2):53–80, 1987.

[Weg87] Ingo Wegener. *The complexity of Boolean functions*. Teubner, 1987.