Hardness of Approximating Σ_2^p Minimization Problems

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

Abstract

We show that a number of natural optimization problems in the second level of the Polynomial Hierarchy are Σ_2^p-hard to approximate to within n^ϵ factors, for specific $\epsilon > 0$. The main technical tool is the use of explicit dispersers to achieve strong, direct inapproximability results. The problems we consider include Succinct Set Cover, Minimum Equivalent DNF, and other problems relating to DNF minimization. Under a slightly stronger complexity assumption, our method gives optimal $n^{1-\epsilon}$ inapproximability results for some of these problems. We also prove inapproximability of a variant of an NP optimization problem, Monotone Minimum Satisfying Assignment, to within an n^ϵ factor using the same technique.

1. Introduction

One of the earliest approximation algorithms for an NP optimization problem was Johnson's greedy algorithm for approximating SET COVER to within a factor of $\ln n$, where n is the number of elements in the ground set [11]. Proving a matching inapproximability result has been significantly more difficult. Since the work of Lund and Yannakakis [14], all PCP reductions to set cover have relied on an auxiliary combinatorial construction (partition systems). Only with the recent improvements in low-error PCP characterizations of NP [22, 3], has approximating SET COVER within a factor of $O(\log n)$ been shown to be NP-hard (although the earlier result of Feige [8] gives an optimal threshold of $(1-\epsilon)\ln n$ under a slightly stronger complexity assumption).

If we allow an *exponentially* large ground set, with *succinctly represented* sets, we get the following Σ_2^p optimization problem:

> SUCCINCT SET COVER: Given a collection S of subsets of $\{0,1\}^n$ whose union is $\{0,1\}^n$, each represented as the on-set of a 3-DNF formula, what is the smallest set cover $S' \subseteq S$?

We would expect (by analogy to SET COVER) that this problem is Σ_2^p-hard to approximate within a factor n^ϵ for some fixed $\epsilon > 0$. Is the PCP machinery (adapted for the class Σ_2^p, as in [12]), together with suitably "scaled up" partition systems, necessary to prove such a result? In this paper, we show that strong inapproximability results for SUCCINCT SET COVER may be obtained using explicit *dispersers*, which are bipartite graphs with certain one-sided random-like properties (the formal definition is in the next section). We also give similar inapproximability results for a number of other natural Σ_2^p optimization problems.

Natural problems in the second level of the hierarchy are few and far between, and there are even fewer natural Σ_2^p *optimization* problems. However, the known optimization problems that do populate this class are important practical problems relating to logic synthesis and circuit minimization. Proving Σ_2^p-completeness for the decision versions of circuit minimization problems is a longstanding open problem that dates to the introduction of the Polynomial Hierarchy [28]. Most variants remain open; however, Σ_2^p-completeness was recently shown for one of the simplest circuit minimization problems, the Minimum Equivalent DNF problem [33]. This problem can be seen as a sort of succinct set cover problem: the well-know exact algorithm for this problem (the Quine-McCluskey algorithm [19, 15]) generates the set of all prime implicants of the input DNF ϕ, and then solves the resulting covering problem; that is, it selects the smallest collection of prime implicants that cover all the ones of ϕ. The generated covering problem may in general be exponentially large; we can therefore think of the input DNF as representing this problem *succinctly*. Indeed, SUCCINCT SET COVER can be reduced to several problems relating to DNF minimization, as we will see in Section 4.

A survey of other problems that have been shown to be Σ_2^p-complete may be found in [24]; some simple Σ_2^p-hard optimization problems have been shown to be hard to approximate in [33] and [34]. Prior to this paper, no formal inapproximability results were known for any natural Σ_2^p op-

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

timization problems, although in [25], deciding the V-C dimension is shown to be Σ_3^p-complete, and some limited (additive) inapproximability results are given. Some positive approximability results for the circuit minimization problem are implied by results in [13, 4].

Dispersers, first defined by Sipser [26], and their two-sided analogues, extractors, have been the focus of much research in recent years (see the survey in [16]), and have found applications in a wide range of areas, including simulating randomized algorithms with weak random sources [37, 27], explicit constructions of expanders, superconcentrators and sorting networks [35, 30], constructive leader election [38, 23], and several diverse applications in complexity theory [18, 2, 29, 26, 10]. The dispersers we use in this paper are ones that work on sources with very small min-entropy; constructions for this parameter range are in [27, 9] (logarithmic degree) and [16, 32, 21] (polylogarithmic degree).

Extractors have been used before to prove hardness of approximation [36, 27]; these methods can be best seen as an amplification of inapproximability results achieved using PCPs. The novelty of our approach is that we use dispersers directly (albeit in a slightly different setting), without PCPs, which leads to simple reductions that may yield insight into structural aspects of the problems that are hard to approximate. In particular, the use of dispersers in this setting validates the intuition that instances of SUCCINCT SET COVER that are hard to approximate should have the property that "candidate" small covers are dispersed among the collection of sets so that the union of even a small number of them is large.

One question raised by this paper is whether these techniques can find applications in proving inapproximability results for certain NP optimization problems. We make initial steps in this direction in Section 6, where we apply our method to a variant of the MONOTONE MINIMUM SATISFYING ASSIGNMENT problem. For many NP optimization problems, strong inapproximability results have already been obtained using PCPs, but alternative or more direct methods have the potential to lead to a better understanding of what makes certain problems hard to approximate.

2. Preliminaries

Approximation. Let Π be a minimization problem. With each instance I of Π, there is an associated set of feasible solutions. Note that for Σ_2^p optimization problems, even determining whether s is a feasible solution requires a coNP computation; therefore any reasonable approximation algorithm for these problems runs in polynomial time *with access to an NP oracle*. (This observation is borne out in practice, where circuit minimization packages make use of a TAUTOLOGY subroutine [6]). The optimum of instance I is defined to be the minimum solution value over all feasible solutions s.

To show that Π is hard to approximate, we consider a promise variant of Π, Π WITH GAP $r(n)$. The positive instances of this problem are those instances I of Π for which $OPT(I)$ is at most some upper bound $B(n)$, where n is the size of I, and the negative instances are those instances I of Π for which $OPT(I) > r(n)B(n)$. If this promise problem is $\mathcal{C}$-complete for some complexity class $\mathcal{C}$, then an approximation algorithm for Π that achieves an approximation ratio of $r(n)$ can distinguish between positive and negative instances of Π WITH GAP $r(n)$, and hence solve any problem in class $\mathcal{C}$. We say then that Π is $\mathcal{C}$-hard to approximate within a factor $r(n)$.

Reductions. All of the reductions in this paper between optimization problems are A-reductions [5] (denoted by $\leq_A$), which preserve approximability within constant factors. Our reductions are also *gap-preserving* in the following sense: they map an instance I of problem Π to an instance I' of problem Π' such that $OPT(I) = \Theta(p(|I|)OPT(I'))$, where p is a fixed polynomial scaling factor. In particular, if Π WITH GAP n^ϵ is Σ_2^p-complete for some $\epsilon > 0$, and Π reduces to Π', then Π' WITH GAP $n^{\epsilon'}$ is also Σ_2^p-complete, for some $\epsilon' > 0$.

Dispersers. We now introduce some basic definitions relating to disperser graphs. In general, given a graph $G = (V, E)$, we denote by $\Gamma(v)$ the set of all neighbors of v in G; if V' is a subset of the vertices, then $\Gamma(V') = \cup_{v \in V'} \Gamma(v)$.

Definition 1 (Disperser) *A* (k, ϵ)-disperser *is a bipartite graph* $G = (U = [2^n], V = [2^m], E)$ *with left-degree* 2^d *with the following property: for every subset* $U' \subseteq U$ *of size at least* 2^k, $|\Gamma(U')| > (1-\epsilon)|V|$. *A disperser is* explicit *if there exists a function* $G(u, i)$ *that computes the ith neighbor of vertex* $u \in U$ *in time polynomial in* n.

Dispersers are the one-sided analogue of extractors, which require that any distribution on the vertices U with *min-entropy* at least k, together with the uniform distribution on $[2^d]$ induces a distribution on V that is ϵ-close to uniform. A distribution X on U has min-entropy k if for all $u \in U$, $\Pr[X = u] \leq 2^{-k}$. Every extractor is a disperser with identical parameters, and in fact all of the dispersers we use in this paper are actually extractors. Table 2 summarizes the parameters for the explicit dispersers used in this paper, along with the optimal, and non-constructive, set of parameters (we assume ϵ is constant).

Implicants. Several of the Σ_2^p optimization problems that we consider are related to finding *implicants* of Boolean functions. Let f be a Boolean function. An implicant of f is a conjunction C of literals that implies f; equivalently, one

reference	min-entropy k	right-hand-side size 2^m	left-degree 2^d
SZ [27, Lemma 3.2]	any k	$m = k + d - O(1)$	$d = 4k + O(\log n)$
RRV [21]	any k	$m = (1 - \alpha)k$, α an arbitrarily small constant	$d = O(\log^2 n)$
non-constructive and optimal [20]	any k	$m = k + d - O(1)$	$d = O(\log n)$

Table 1. Parameters for dispersers used in this paper.

can think of an implicant of f as a partial assignment to f that forces it to 1. We often think of conjunctions (and implicants in particular) as sets of literals. Using this interpretation, we can make the following elementary observation: if C_1 and C_2 are conjunctions with $C_1 \subseteq C_2$, and C_1 is an implicant of f, then C_2 is also an implicant of f. An implicant C is a *prime implicant* if the only $C' \subseteq C$ that is an implicant of f is C itself. An assignment A to f is *consistent* with a conjunction C if no variable is set to contradictory values in A and C. We say that a conjunction C *covers* every assignment consistent with C. An *essential prime implicant* is an implicant that covers some assignment that is covered by no other prime implicant.

Quantified SAT. The basic Σ_2^p-complete problem that we reduce from is QSAT$_2$. An instance of QSAT$_2$ is given by a 3-DNF formula $\phi(x, y)$, where x and y are sets of variables. The instance is a positive instance if and only if $\exists x \forall y \quad \phi(x, y)$. Notice that this is equivalent to asking whether there exists an implicant of ϕ consisting of exactly the x variables.

Nondeterministic circuits. The problem in our main reduction accepts as its input a description of a *co-nondeterministic circuit*, or *co-ND circuit* for short. A *nondeterministic circuit* (*ND circuit*) is a circuit whose gates are the usual AND, OR, NOT, and fan-in zero INPUT gates, plus fan-in zero CHOICE gates. The circuit accepts an input iff there exists some truth assignment to the CHOICE gates on which the circuit outputs 1. The only difference between an ND circuit and a co-ND circuit is the acceptance criterion: a co-ND circuit accepts an input iff for all truth assignments to the CHOICE gates, the circuit outputs 1.

2.1. Outline

In section 3 we present the basic reduction, establishing that our first Σ_2^p optimization problem is Σ_2^p-hard to approximate within an n^ϵ factor. In section 4 we give approximation preserving reductions to a number of other problems in the second level of the hierarchy, establishing Σ_2^p-hardness of approximating them within n^ϵ as well. In section 5 we introduce an improvement to the basic reduction, and give a more detailed analysis of the gaps achieved. For the basic problem, we obtain a gap of $n^{1/5-\epsilon}$ for any $\epsilon > 0$, using currently best disperser constructions; this can be improved to an optimal $n^{1-\epsilon}$ if certain parameters of the disperser can be improved. Finally, we show that the optimal $n^{1-\epsilon}$ gap *is* achieved under a strengthened complexity assumption. In section 6 we apply our technique to an NP optimization problem, achieving a gap of n^ϵ without using PCPs. We end with conclusions and open problems in section 7.

3. The main reduction

Our basic Σ_2^p optimization problem is closely related to SUCCINCT SET COVER, and makes for a simpler exposition of the main reduction:

> MONOTONE MINIMUM WEIGHT WORD (MMWW): Given a co-ND circuit C that accepts exactly a non-empty monotone set, what is the string accepted by C with the fewest number of ones?

Here, a *monotone set* is a subset $A \subseteq \{0,1\}^n$ with the property that if $x \in A$, then $x' \in A$ for all x' obtained by flipping one or more bits in string x from 0 to 1. In reductions, we will describe an instance of MMWW by describing a coNP algorithm. Using standard methods, we can construct the required co-ND circuit from this algorithm, and notice that the size of the constructed circuit will be polynomial in the input size. Also, we often identify the input $x \in \{0,1\}^n$ with a subset of some underlying set S of size n, where the ones in the input string select the elements of S that are in the subset. Under this interpretation, monotonicity requires that if the circuit accepts input $S' \subseteq S$ then it must accept all $S'' \supseteq S'$. The "minimum weight word" is the smallest subset $S' \subseteq S$ that the circuit accepts. It is easy to see that the decision version of MMWW is Σ_2^p-complete when it is viewed in this way. We describe a reduction below.

A simple reduction. Given an instance of QSAT$_2$: $\phi(x, y)$ with $|x| = n$, let S be the set of $2n$ literals

$\{x_1, \overline{x_1}, x_2, \overline{x_2}, \ldots, x_n, \overline{x_n}\}$. The coNP algorithm accepts an input $S' \subseteq S$ if $|S'| \geq n+1$, or if S' contains exactly one literal for each i, and the implied assignment X to the x variables satisfies the QSAT_2 instance; otherwise it rejects. The reader may verify that the set of strings accepted by this algorithm is non-empty and monotone, as required, and that the minimum weight string accepted by the algorithm has weight $\leq n$ if and only if the instance of QSAT_2 is a positive instance.

This reduction shows that MMWW WITH GAP $1 + r(n)$ is Σ_2^p-complete for a negligible $r(n)$. Our goal in the next section will be to improve that gap dramatically using dispersers.

3.1. Dispersers and efficient decoding

We will use the same method as in the simple reduction in the previous section: certain subsets of the underlying set S correspond to possible solutions to the QSAT_2 instance. However, our task is complicated by the monotonicity requirement. In order to achieve a nontrivial gap $r(n)$, we need to describe a coNP algorithm that performs a seemingly difficult (for coNP) computational task: given an input $S' \subseteq S$ with $|S'| \leq B(n)r(n)$, the algorithm must determine if there exists some subset $S'' \subseteq S'$ (of size at least $B(n)$) that it *would* accept. Even for constant $r(n)$, there are an exponential number of such subsets, since we expect $B(n)$ to be polynomial in the size of the instance. We therefore need to limit the number of such subsets that must be checked, and we need the coNP algorithm to be able to efficiently "decode" the relevant subsets from the input S'.

Dispersers provide a natural solution to the first goal: if the underlying set S is the right-hand-side of a disperser, and the subsets are the neighbor sets of each of the vertices on the left-hand-side, then a subset $S' \subseteq S$ of "reasonable" size (i.e., a constant fraction of $|S|$) cannot contain too many subsets. To actually find these subsets, we need a slightly more complicated encoding, described in the proof of the next theorem.

Theorem 1 MMWW WITH GAP N^ϵ *is* Σ_2^p*-complete for some* $\epsilon > 0$*, where* N *is the size of the instance.*

Proof: The reduction is from QSAT_2. We are given an instance of QSAT_2: $\phi(x, y)$ with $|x| = n$. We use the SZ construction to obtain a $(k, 1/2)$-disperser $G = (U = [2^{n+1}], V = [2^{k+d-O(1)}], E)$ with $k = c \log n$, for some constant c to be specified later. Our choice of $1/2$ is unimportant – any constant less than 1 will suffice.

We think of the vertices in U as being organized into a full binary tree with 2^n leaves corresponding the 2^n possible assignments to the x variables of the QSAT_2 instance. By an abuse of notation, we identify these leaves with the assignments. For each assignment X to the x variables, define the set $V_X \subseteq V$ to be $\Gamma(P_X)$, where P_X is the set of vertices along the unique path from the root to leaf X.

Given a subset $V' \subseteq V$, we say that a vertex $u \in U$ is *active* if $\Gamma(u) \subseteq V'$. By the disperser property of G, there are at most 2^k active vertices with respect to any subset V' of size at most $|V|/2$. Given a subset V' within this size limit, we need to be able to identify the set of all V_X contained within it ("decode" V'). This is accomplished with the decoding procedure described below, which performs a breadth-first-search from the root of the tree, pruning the search whenever a non-active vertex is encountered:

Decode(V')
  Precondition: $V' \subseteq V$ *and* $|V'| \leq |V|/2$
  Returns: $\{X : V_X \subseteq V'\}$
  if $\Gamma(\text{root}) \not\subseteq V'$ then return $\{\}$
  else **active** $\leftarrow$ root
  for $i \leftarrow 1$ to n do
      active $\leftarrow \{u \in U : u$ is a child of some $u' \in$ **active**
              and $\Gamma(u) \subseteq V'\}$
  return **active**

The total running time of this procedure is $O((\text{time to compute } \Gamma(u)) \times (\#\text{ active vertices}))$. Since 2^k, 2^d, and $|V|$ are polynomial in n, the running time is polynomial.

We now describe the instance of MMWW by giving a coNP algorithm A, whose input is interpreted as a subset of the underlying set V. On input $V' \subseteq V$, A accepts if $|V'| > |V|/2$; otherwise, A accepts if and only if $\exists X \in \text{Decode}(V') \forall y\ \phi(X, y)$. To see that this runs in coNP, we need the following claim:

Claim 1 *Given formula* $\phi(x, y)$*, for any set* S *of assignments to the* x *variables with* $|S| \leq \ell$*, we have:*

$$\exists X \in S\ \forall y\ \phi(X, y) \Leftrightarrow \forall y_1, \ldots, y_\ell \bigvee_{X \in S} \wedge_{i=1}^{\ell} \phi(X, y_i).$$

Proof: First, if there exists an assignment $X \in S$ for which $\forall y \quad \phi(X, y) = 1$, then for that X, we have $\forall y_1, y_2, \ldots y_\ell \wedge_{i=1}^{\ell} \phi(X, y_i) = 1$ and hence the right-hand-size is true.

Now, suppose there is no assignment $X \in S$ for which $\forall y\ \phi(X, y)$. Then for each $X_i \in S$, there exists an assignment Y_i to the y variables such that $\phi(X_i, Y_i) = 0$. Therefore, for every $X \in S$, $\wedge_{i=1}^{\ell} \phi(X, Y_i) = 0$, so for the assignment in which $y_i = Y_i$ for all i, the right-hand-side is false. ■

Using the claim, the set L of strings accepted by A can be described in the following way, from which it is clear that A runs in coNP:

$$L = \{V' \subseteq V : \forall y_1, y_2 \ldots y_{2^k}\ |V'| > |V|/2\ \vee \\ \vee_{X \in \text{Decode}(V')} \wedge_{i=1}^{2^k} \phi(X, y_i)\}$$

Let us verify monotonicity: we claim that for every $V' \subseteq V$ that A accepts, it also accepts all $V'' \supseteq V'$. Let V' be a subset that A accepts. It must be that either $|V'| > |V|/2$, in which case the claim follows easily, or there exists some assignment X that satisfies the instance of QSAT$_2$ for which $V_X \subseteq V'$. Then, for all $V'' \supseteq V'$, either $|V''| > |V|/2$ and A accepts, or else decoding is performed and $X \in \text{Decode}(V'')$, which makes A accept, as required. Finally, notice that A accepts the all-ones string, as required.

We now show that the gap is N^ϵ for some $\epsilon > 0$. If the instance of QSAT$_2$ is a positive instance, then some set V_X of size at most nD is accepted by A, where $D = 2^d$ is the left-degree of G. In the negative case, no set V' of size at most $|V|/2$ causes A to accept. This leads to a gap of $|V|/2nD = \Omega(n^{c-1})$. The size N of the co-ND circuit generated by this reduction is polynomial in n, and the N^ϵ gap follows. ∎

4. The problems

In this section we give reductions from MMWW to a number of natural minimization problems in the second level of the hierarchy, including SUCCINCT SET COVER, defined in the introduction. The first two problems are variants of the DNF minimization problem; note that the reduction we give for the second variant is the first to prove even that the decision version is Σ_2^p-complete.

> MIN DNF: Given a DNF formula ϕ, what is the equivalent DNF formula ϕ' with the fewest occurrences of literals?

> MIN TERM DNF: Given a DNF formula ϕ, what is the equivalent DNF formula ϕ' with the fewest terms?

The next two problems are sub-steps in widely used heuristics for DNF minimization. Despite their hardness, as demonstrated in this paper, these two problems (more often the second) are sometimes solved *exactly* in DNF minimization packages [6]:

> SHORTEST IMPLICANT CORE: Given a DNF formula ϕ and an implicant C of ϕ, what is the shortest implicant $C' \subseteq C$?

> IRREDUNDANT: Given a DNF formula $\phi = t_1 \vee t_2 \vee \cdots \vee t_n$, what is the smallest cardinality index set $I \subseteq \{1 \ldots n\}$ for which $\phi \equiv \bigvee_{i \in I} t_i$?

Theorem 2 *The gap versions of the following problems:* SUCCINCT SET COVER, MIN DNF, MIN TERM DNF, SHORTEST IMPLICANT CORE, *and* IRREDUNDANT *are* Σ_2^p*-complete, with gap* N^ϵ *for some* $\epsilon > 0$*, where* N *is the size of the instance.*

Proof: We give two of the reductions (for SUCCINCT SET COVER and MIN TERM DNF) in some detail; the others are either easy or similar to one of these two.

MMWW $\leq_A$ SUCCINCT SET COVER**.** Let co-ND circuit C be an instance of MMWW, and let $x_1, x_2, \ldots x_m$ be the input variables for C. Using standard methods, we obtain a 3-DNF formula ψ from circuit C with the following properties: (1) the variables of ψ are $x_1, x_2, \ldots x_m$ plus a linear (in the size of C) number of other variables z; and (2) for all assignments X to the x variables, $\forall z \ \psi(X, z)$ holds if and only if X is accepted by C. We will have $m + 1$ sets in our instance of SUCCINCT SET COVER. The first set is given by ψ. The remaining m sets are specified by the single literal $\overline{x_i}$ for $i = 1 \ldots m$.

Claim 2 *Circuit* C *accepts a string* X^* *of weight* k *if and only if the instance of* SUCCINCT SET COVER *has a cover of size* $k + 1$.

Proof: ($\Rightarrow$) We claim that ψ together with the k sets $\overline{x_i}$ for each x_i set to true in X^* constitute a cover. Let X, Z be an assignment to the x and z variables of ψ. Then either (1) the set encoded by X contains X^*, in which case $\psi(X, Z) = 1$ by monotonicity, or (2) there exists some x_i set to zero by X and one by X^*, in which case $\overline{x_i}$ is true. This holds for all assignments X, Z, so the collection of $k + 1$ sets is indeed a cover.

($\Leftarrow$) We have a cover of size $k + 1$. This cover must include ψ, which is the only set that covers the all-ones assignment. Let X^* be the assignment to the x variables that sets x_i to one if and only if the set $\overline{x_i}$ is in the cover. We claim that C accepts X^*. If not, then there exists some assignment Z to the z variables for which $\psi(X^*, Z) = 0$. But this assignment is not in the on-set of any of the sets in the cover, a contradiction. ∎

MMWW $\leq_A$ SHORTEST IMPLICANT CORE**.** We simply take the DNF formula ψ in the reduction above together with the implicant $x_1 x_2 \ldots x_m$ as our instance of SHORTEST IMPLICANT CORE.

SHORTEST IMPLICANT CORE $\leq_A$ MIN TERM DNF**.** Let DNF ϕ and implicant X of ϕ be an instance of SHORTEST IMPLICANT CORE. Let $s_1, s_2, \ldots s_m$ be the terms of $(\phi \wedge \neg X)$ after it is multiplied out into DNF. The instance of MIN TERM DNF is given by:

$$\phi' = \bigvee_{i=1}^{m} s_i' \vee \bigvee_{x_i \in X} \bigvee_{j=1}^{m} u_i^j,$$

where $s_i' = s_i z_1 z_2 \ldots z_{i-1} z_{i+1} \ldots z_m$ and $u_i^j = \overline{x_i} a_j z_1 z_2 \ldots z_m$ and the z_i and a_j are new variables.

Claim 3 *There exists an implicant $X' \subseteq X$ of ϕ of length k if and only if ϕ' has an equivalent DNF ϕ'' with at most $m(k+1)$ terms.*

Proof: ($\Rightarrow$) Suppose $X' \subseteq X$ is an implicant of ϕ of length $|X'| = k$. We construct ϕ'' as follows:

$$\phi'' = \bigvee_{i=1}^{m} s'_i \vee \bigvee_{x_i \in X'} \bigvee_{j=1}^{m} u_i^j \quad (1)$$

Since ϕ'' includes a subset of the terms of ϕ', to prove equivalence, we only need to verify that ϕ'' is true on all assignments for which ϕ' is true. Let A be an assignment for which $\phi'(A) = 1$. If A has at least one z variable false, or all a variables false, then it must be consistent with some s'_i, since the only such assignments covered by ϕ' are those covered by the s'_i. Otherwise, A has all z variables true and at least one a variable true. Now, it can be argued that if A is consistent with X', then it is covered by the s'_i; otherwise, it is covered by some u_i^j included in ϕ''.

($\Leftarrow$) Suppose ϕ'' is a DNF equivalent to ϕ'. Our goal is to construct a short implicant of ϕ from ϕ''. We make two initial observations. First, if t is a term of ϕ'' that includes *no* true a variables, then $t \supseteq s'_i$ for some i. Second, there must be at least m terms t of ϕ'' for which $t \supseteq s'_i$ for some i. We omit the fairly easy proof of these observations in this version of the paper. As a first step, we discard all terms t of ϕ'' for which $t \supseteq s'_i$ for some i. By the second observation above, this accounts for at least m terms.

We now restrict our attention to the R remaining terms of ϕ''. By the first observation above, these remaining terms all include at least one positive a variable. Since all assignments covered by ϕ' have some x_i set to false, each of these terms also contains some negated x variable. We label each of these terms with a pair (i, j), where j is the index of one of the positive a variables in the term and i is the index of one of the negated x variables in the term (in each case if there are more than one, we pick arbitrarily). By a pigeonhole argument, there must be some j for which no more than $\lceil R/m \rceil$ (i, j)-labeled terms are present. For that value of j, we include in X' all literals x_i for which an (i, j)-labeled term is present.

We claim that X' is an implicant of ϕ. If it is not, then there exists some assignment A consistent with X' for which ϕ is false. Since X is an implicant of ϕ, this A must be inconsistent with X, and hence A sets some x variable to false. Extend assignment A so that it sets all z variables to true, and all a variables to false *except* a_j for the value of j selected above, which it sets to true. Formula ϕ' is true on the resulting assignment, because ϕ' covers all assignments with some x variable false, all z variables true, and at least one a variable true. However, by construction, the assignment is inconsistent with all s'_i (since $\phi(A) = 0$, $(\phi \wedge \neg X) = 0$, and therefore A is inconsistent with all s_i). This implies that it is inconsistent with all of the discarded terms. Furthermore, it is inconsistent with all of the R remaining terms t, because it either sets some a variable to false that appears positively in term t, or else a_j appears positively in term t and the assignment sets some x_i to true that is false in term t. Therefore ϕ'' is false on this assignment, but ϕ' is true, a contradiction.

The size of X' is $\lceil R/m \rceil$, where $R \leq |\phi''| - m$, so if k is the smallest integer for which $|\phi''| \leq m(k+1)$, then $|X'| \leq k$, as required. ■

For IRREDUNDANT and MIN DNF, we use essentially the same reduction as we did for MIN TERM DNF. For IRREDUNDANT, note that the formula given in (1) consists only of terms from the input DNF ϕ', and that the second half of the proof above becomes easier when the equivalent DNF has only a subset of the terms from the input DNF. For MIN DNF, note that the lengths of terms in the formula given in (1) are at most $m + 3$ and in the second half of the proof above, the terms must all have length at least m. ■

In the next section, we give a tighter analysis of the gaps for these problems, specifying them in terms of parameters of the disperser used in the underlying reduction.

5. Improvements and generalizations

In this section we show that the gap for MMWW obtained in Section 3 can be "boosted" by composing the input co-ND circuit with itself (this is similar to the "self-improvement" technique of [1]). The exact gap achieved is then a simple function of the parameters of the underlying disperser. We give the exact gaps obtainable using the best current disperser constructions, noting that for optimal dispersers, the gap is optimal ($n^{1-\epsilon}$ for any $\epsilon > 0$) up to lower order terms. We give similar exact gaps for the other problems from Section 4. Finally, we show that optimal gaps *are* achievable if the complexity assumption is strengthened slightly, using dispersers with poly-logarithmic degree.

5.1. Composing circuits to improve the gap

Definition 2 *Let C be a co-ND circuit. We define C^ℓ for $\ell \geq 1$ recursively as follows: $C^1 = C$ and C^ℓ is $C^{\ell-1}$ with independent copies of C inserted at each of its inputs.*

Lemma 3 *Let C be a co-ND circuit with n inputs that accepts exactly a monotone set of strings. The following hold:*

1. $|C^\ell| \leq n^\ell |C|$
2. *C accepts a string of weight k if and only if C^ℓ accepts a string of weight k^ℓ.*
3. *the set of strings accepted by C^ℓ is monotone.*

Proof: For C^2, we need one copy of the original co-ND circuit C, plus n additional copies for each of the inputs. For C^3, we need n^2 additional copies; and so on. In general the size of C^ℓ is $(1+n+n^2+\ldots+n^{\ell-1})|C|$, which is at most $n^\ell|C|$.

For the second item, the forward direction is simple: if X is a string of weight k accepted by C, then we force the appropriate k copies of $C^{\ell-1}$ to output one so that the induced input to the topmost copy of C is X. By induction, each copy of $C^{\ell-1}$ can be made to output one with an input of weight $k^{\ell-1}$, so the total weight of the accepted string is k^ℓ.

In the other direction, suppose C^ℓ accepts a string X with weight k^ℓ. At the bottom level, C^ℓ has $n^{\ell-1}$ copies $C_1, C_2, \ldots C_{n^{\ell-1}}$ of the original circuit C. Consider the inputs $X_1, X_2, \ldots X_{n^{\ell-1}}$ to each of these copies, obtained by projecting X onto the appropriate subset of the input variables. If for some i, C_i rejects string X_i, then by monotonicity we can safely set all of the inputs to that C_i to false without changing the value of the output of C^ℓ'. So we may assume without loss of generality that each X_i is either the all-zeros string or it is accepted by C_i. So either some X_i has weight at most k and we are done, or else X induces an assignment to the inputs of $C^{\ell-1}$ with weight at most $k^{\ell-1}$, and we are done by induction on ℓ.

For the final item, we note that the composition of co-ND circuits that accept a monotone set is a co-ND circuit that accepts a monotone set. ■

We use this lemma in the next section to transform an instance of MMWW so that its size is dominated by the number of deterministic variables, instead of the co-ND circuit size required in the reduction of Theorem 1.

5.2. Exact gaps

Using Lemma 3 from the previous section, we obtain the following relationship between disperser parameters and the gap achievable for MMWW using the main reduction:

Theorem 4 *Let $G = (U = [2^{n+1}], V = [2^m], E)$ be an explicit $(k, 1/2)$-disperser with left-degree 2^d and $\Omega(\log n) \le k, d \le m$, and define $r = 1 - \frac{d+\log n}{m}$. Then we can reduce any instance of* QSAT$_2$ *to an instance of* MMWW *with size $N \le poly(2^m)$ and gap $N^{r-\epsilon}$ for any constant $\epsilon > 0$.*

Proof: First, we follow the proof of Theorem 1. This produces from an instance of QSAT$_2$ a co-ND circuit C whose size is polynomial in 2^m, 2^k, 2^d and n, which is at most 2^{mp} for some constant p, and the number of input variables is 2^m. The gap is $\Omega(2^{m-d-\log n})$. Choose $\ell = p\epsilon^{-1}$. By Lemma 3, the size of circuit C^ℓ is at most $N = 2^{m\ell+mp}$, and the gap for C^ℓ is $\Omega(2^{m-d-\log n})^\ell$. We claim that this gap is larger than $N^{r-\epsilon}$. This follows from:

$$\begin{aligned}(m\ell+mp)(r-\epsilon) &= (\ell+p)(m-d-\log n)-(1+\epsilon)mp \\ &< \ell(m-d-\log n-O(1))\end{aligned}$$

which holds for sufficiently large m. ■

A closer analysis of the reductions in the proof of Theorem 2 yields the following corollary:

Corollary 5 *If* MMWW WITH GAP $N^{r-\epsilon}$ *is Σ_2^p-complete for some constant r and for all $\epsilon > 0$, then the gap versions of* SHORTEST IMPLICANT CORE *and* SUCCINCT SET COVER *are Σ_2^p-complete with gap $N^{r-\epsilon}$ for all $\epsilon > 0$, and the gap versions of* MIN TERM DNF, MIN DNF, *and* IRREDUNDANT *are Σ_2^p-complete with gap $N^{r/4-\epsilon}$ for all $\epsilon > 0$.*

Finally, plugging the best (for our purposes) currently known explicit disperser into Theorem 4, we get the following result:

Theorem 6 *The gap versions of problems* MMWW, SHORTEST IMPLICANT CORE, *and* SUCCINCT SET COVER *are Σ_2^p-complete with gap $N^{1/5-\epsilon}$ for any $\epsilon > 0$, where N is the size of the instance. The gap versions of problems* MIN TERM DNF, MIN DNF, *and* IRREDUNDANT *are Σ_2^p-complete with gap $N^{1/20-\epsilon}$ for any $\epsilon > 0$.*

Proof: We prove the stated gap for MMWW; the gaps for the remaining problems follow from Corollary 5. We plug the SZ disperser into Theorem 4. Pick $k = c\log n$. Then $d = (4c+O(1))\log n$ and $m = (5c+O(1))\log n - O(1)$. Since $m = O(\log n)$, the size of the generated instance of MMWW is polynomial in n. For c large enough, $r = 1 - \frac{d+\log n}{m} > 1/5 - \epsilon/2$; the gap of $N^{1/5-\epsilon}$ follows. ■

It is interesting to note that with an optimal disperser the gap achieved for MMWW via Theorem 4 is $N^{1-\epsilon}$, which is optimal up to lower order terms.

5.3. Optimal gaps under a stronger complexity assumption

If we use a disperser with polylogarithmic degree, we can achieve the optimal $N^{1-\epsilon}$ gap for MMWW, together with the improvements this implies for the other problems. The drawback is that the reduction produces an instance of size $n^{O(\log n)}$, so the complexity assumptions must be strengthened accordingly. The best available disperser for this purpose is due to Raz, Reingold, and Vadhan [21], improving on previous work of Trevisan [32]. In the following theorem, $\tilde{\mathrm{P}}$ denotes quasipolynomial time, i.e., $\tilde{\mathrm{P}} = \mathrm{TIME}(n^{O(\log n)})$.

Theorem 7 *Assume $\Sigma_2^p \not\subseteq \tilde{P}^O$, for some oracle O. Then there is no $N^{1-\epsilon}$-approximate algorithm for* MMWW *that runs in polynomial time with access to oracle O, for any $\epsilon > 0$.*

Proof: Plug the RRV disperser into Theorem 4. Pick $k = c\log^2 n$. Picking α to be, say, $1/2$, we have $m = (1/2)c\log^2 n$, and $d = O(\log^2 n)$. The size of the generated instance of MMWW is quasipolynomial in n. For c large enough, $r = 1 - \frac{d+\log n}{m} > 1 - \epsilon/2$; the gap of $N^{1-\epsilon}$ follows. ■

6. Hardness of approximation for an NP optimization problem

The following problem is the "NP version" of the MMWW problem:

> MONOTONE MINIMUM SATISFYING ASSIGNMENT: Given a (deterministic) circuit C that accepts exactly a non-empty monotone set, what is the satisfying assignment with the fewest number of ones?

Several special cases of this problem have been considered previously; to our knowledge, we are the first to propose this very general version. A variant in which the input circuit is also required to be a monotone circuit, i.e., a circuit over the monotone basis $\{\wedge, \vee\}$, is related to NP optimization problems concerning minimum propositional proof length [1]. In [7], they show that the variant in which the input circuit is required to be a depth- three $\{\wedge, \vee\}$-formula is NP-hard to approximate within a $2^{\log^{1-o(1)} N}$ factor. In this section we show that the very general version of this problem stated above is NP-hard to approximate within a factor of $N^{1/5-\epsilon}$, for any $\epsilon > 0$. As in the previous section, this factor can be improved to an optimal $N^{1-\epsilon}$ if optimal explicit dispersers exist, and separately, under a strengthened complexity assumption.

Theorem 8 MONOTONE MINIMUM SATISFYING ASSIGNMENT *is NP-hard to approximate within a factor of $N^{1/5-\epsilon}$, for any $\epsilon > 0$, where N is the size of the instance.*

Proof: We use the same technique as in the proof of Theorem 1 together with analysis similar to that in the proof of Theorem 6. We use a $(k, 1/2)$-disperser $G = (U = [2^{n+1}], V = [2^m], E)$, with left-degree $D = 2^d$. Given an instance $\phi(x)$ of SAT, we construct a polynomial-size deterministic circuit that accepts exactly the following set L of strings (which we interpret as subsets of the underlying set V, as before):

$$L = \{V' \subseteq V : |V'| > |V|/2 \vee \bigvee_{X \in \text{Decode}(V')} \phi(X)\}$$

Note that Lemma 3 applies to ordinary deterministic circuits, and so a version of Theorem 4 holds. Using the SZ disperser, and choosing $k = c\log n$, we have $d = (4c + O(1))\log n$ and $m = (5c + O(1))\log n - O(1)$, as in the proof of Theorem 6. For sufficiently large c, we obtain a gap of size $N^{1/5-\epsilon}$. ■

As above, we can use the RRV disperser to obtain a better gap under a stronger complexity assumption:

Theorem 9 *Assume NP $\not\subseteq \tilde{P}$. Then there is no polynomial-time algorithm that approximates* MONOTONE MINIMUM SATISFYING ASSIGNMENT *within a factor of $N^{1-\epsilon}$.*

7. Conclusions and open problems

The problems in this paper represent a particularly striking example of the disparity between worst-case analysis and real-world performance. The results in this paper suggest that these problems should be spectacularly hard to deal with, given that they are so hard to approximate *even by algorithms with access to an NP oracle*. However, the logic synthesis community routinely deals with real-world instances of many of these problems quite successfully [6]. An understanding of this phenomenon (what special cases are easier; why certain heuristics perform well on average) is perhaps the most important outstanding challenge here, and in many other settings.

It is likely that MIN DNF, MIN TERM DNF and IRREDUNDANT are in fact Σ_2^p-hard to approximate to within an $n^{1-\epsilon}$ factor. To achieve such a result using the techniques in this paper, tighter reductions from SHORTEST IMPLICANT CORE are needed. There are also several other circuit minimization problems (e.g., Minimum Equivalent Formula, Minimum Equivalent Circuit) that are likely to be at least as difficult as MIN DNF for which even a proof that the decision version is Σ_2^p-complete remains to be found.

We note that even modest improvements in certain paramaters of explicit dispersers would lead to improved, and even optimal, inapproximability ratios for several of the problems in this paper. For example, with respect to the SZ construction [27], it would be sufficient to reduce the dependence of d on k from $4k$ to αk, for arbitrarily small α.

Finally, it would be interesting to see if dispersers can be used to prove inapproximability for NP optimization problems other than the one in Section 6.

Acknowledgements We wish to thank Christos Papadimitriou for many useful discussions.

References

[1] M. Alekhnovich, S. Buss, S. Moran, and T. Pitassi. Minimum propositional proof length is NP-hard to linearly approximate. In *Mathematical Foundations of Computer Science, 23rd International Symposium (MFCS 98)*, pages 176–184, Brno, Czech Republic, August 1998. Springer-Verlag. To appear in Journal of Symbolic Logic.

[2] A. E. Andreev, A. E. F. Clementi, and J. D. P. Rolim. Worst-case hardness suffices for derandomization: A new method for hardness-randomness trade-offs. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Automata, Languages and Programming, 24th International Colloquium (ICALP 97)*, volume 1256 of *Lecture Notes in Computer Science*, pages 177–187, Bologna, Italy, 7–11 July 1997. Springer-Verlag.

[3] S. Arora and M. Sudan. Improved low degree testing and its applications. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC 97)*, pages 485–495, El Paso, Texas, 4–6 May 1997.

[4] N. H. Bshouty, R. Cleve, R. Gavaldà, S. Kannan, and C. Tamon. Oracles and queries that are sufficient for exact learning. *Journal of Computer and System Sciences*, 52(3):421–433, June 1996.

[5] P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Information and Computation*, 93(2):241–262, Aug. 1991.

[6] S. Devadas, A. Ghosh, and K. Keutzer. *Logic Synthesis*. McGraw-Hill, Inc., 1994.

[7] I. Dinur and S. Safra. On the hardness of approximating label cover. Technical Report TR99-015, Electronic Colloquium on Computational Complexity, 1999.

[8] U. Feige. A threshold of $\ln n$ for approximating set cover (preliminary version). In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC 96)*, pages 314–318, Philadelphia, Pennsylvania, 22–24 May 1996.

[9] O. Goldreich and A. Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures and Algorithms*, 11(4):315–343, 1997.

[10] O. Goldreich and D. Zuckerman. Another proof that $BPP \subseteq PH$ (and more). Technical Report TR97-045, Electronic Colloquium on Computational Complexity, 1997.

[11] D. S. Johnson. Approximation algorithms for combinatorial problems. In *Conference Record of Fifth Annual ACM Symposium on Theory of Computing*, pages 38–49, Austin, Texas, 30 Apr.–2 May 1973.

[12] M. Kiwi, C. Lund, A. Russell, D. Spielman, and R. Sundaram. Alternation in interaction. In *Proceedings of the Ninth Annual Structure in Complexity Theory Conference*, pages 294–303, Amsterdam, The Netherlands, 28 June–1 July 1994. IEEE Computer Society Press.

[13] J. Köbler and O. Watanabe. New collapse consequences of NP having small circuits. *SIAM Journal on Computing*, 28(1):311–324, 1998.

[14] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981, Sept. 1994.

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

[16] N. Nisam and A. Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 58(1):148–173, Feb. 1999. Preliminary versions in [31] and [17].

[17] N. Nisan. Extracting randomness: How and why: A survey. In *Proceedings, Eleventh Annual IEEE Conference on Computational Complexity*, pages 44–58, Philadelphia, Pennsylvania, 24–27 May 1996. IEEE Computer Society Press.

[18] N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, Feb. 1996.

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

[20] J. Radhakrishnan and A. Ta-Shma. Tight bounds for depth-two superconcentrators. In *38th Annual Symposium on Foundations of Computer Science (FOCS 97)*, pages 585–594, Miami Beach, Florida, 20–22 Oct. 1997.

[21] R. Raz, O. Reingold, and S. Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC 99)*, 1999.

[22] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC 97)*, pages 475–484, El Paso, Texas, 4–6 May 1997.

[23] A. Russell and D. Zuckerman. Perfect information leader election in $\log^* n + O(1)$ rounds. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS 98)*, pages 576–583, 1998.

[24] M. Schaefer. Graph Ramsey theory and the polynomial hierarchy. Technical Report TR-98-04, University of Chicago, 1998.

[25] M. Schaefer. Deciding the Vapnik-Cervonenkis dimension is Σ_3^p-complete. *Journal of Computer and System Sciences*, 58(1):177–182, Feb. 1999.

[26] M. Sipser. Expanders, randomness, or time versus space. *Journal of Computer and System Sciences*, 36(3):379–383, June 1988.

[27] A. Srinivasan and D. Zuckerman. Computing with very weak random sources. In *35th Annual Symposium on Foundations of Computer Science (FOCS 94)*, pages 264–275, Santa Fe, New Mexico, 20–22 Nov. 1994. IEEE.

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

[29] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. In *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC 99)*, 1999.

[30] A. Ta-Shma. On extracting randomness from weak random sources (extended abstract). In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC 96)*, pages 276–285, Philadelphia, Pennsylvania, 22–24 May 1996.

[31] A. Ta-Shma. Almost optimal dispersers. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC 98)*, pages 196–202, Dallas, Texas, May 1998.

[32] L. Trevisan. Construction of extractors using pseudorandom generators. In *Proceedings of the 31st ACM Symposium on Theory of Computing (STOC 99)*, 1999.

[33] C. Umans. The minimum equivalent DNF problem and shortest implicants. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS 98)*, pages 556–563, 1998.

[34] C. Umans. On the complexity and inapproximability of shortest implicant problems. In *Automata, Languages and Programming, 26th International Colloquium (ICALP 99)*, volume 1644 of *Lecture Notes in Computer Science*, pages 687–696, Prague, Czech Republic, July 1999. Springer-Verlag.

[35] A. Wigderson and D. Zuckerman. Expanders that beat the eigenvalue bound: Explicit construction and applications. In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing (STOC 93)*, pages 245–251, San Diego, California, 16–18 May 1993.

[36] D. Zuckerman. On unapproximable versions of *NP*-complete problems. *SIAM J. Comput.*, 25(6):1293–1304, Dec. 1996.

[37] D. Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16(4/5):367–391, Oct./Nov. 1996.

[38] D. Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11(4):345–367, 1997.