ELSEVIER Discrete Applied Mathematics 66 (1996) 205–218

DISCRETE APPLIED MATHEMATICS

On the geometric separability of Boolean functions[1]

Tibor Hegedűs[a], Nimrod Megiddo[b,c,*,2]

[a] *Department of Computer Science, Comenius University, 84215 Bratislava, Slovakia*
[b] *IBM Almaden Research Center, 650 Harry Road, San José, CA 95120-6099, USA*
[c] *School of Mathematical Sciences, Tel Aviv University, Tel Aviv, Israel*

Received 9 February 1993; revised 26 October 1994

Abstract

We investigate the complexity of the MEMBERSHIP problem for some geometrically defined classes of Boolean functions, i.e., the complexity of deciding whether a Boolean function given in DNF belongs to the class. We give a general argument implying that this problem is co-NP-hard for any class having some rather benign closure properties. Applying this result we show that the MEMBERSHIP problem is co-NP-complete for the class of linearly separable functions, threshold functions of order k (for any fixed $k \geqslant 0$), and some binary-parameter analogues of these classes. Finally, we obtain that the considered problem for unions of $k \geqslant 3$ halfspaces is NP-hard, co-NP-hard and belongs to Σ_2^p, and that the optimal threshold decomposition of a Boolean function as a union of halfspaces cannot even be efficiently approximated in a very strong sense unless $P = NP$. In some cases we improve previous hardness results on the considered problems.

1. Introduction

The class of linearly separable functions corresponds to concepts representable by a single linear threshold (McCulloch–Pitts) neuron — the basic component of neural networks. The problem of recognizing whether a Boolean function is linearly separable (and if yes, constructing its threshold representation) is known as the synthesis problem of threshold logic [18]. This problem can be formulated as follows.
LINEAR SEPARABILITY
Instance: A Boolean function f in DNF.
Question: Is f linearly separable?

* Corresponding author.
[1] A preliminary version of this paper appeared as IBM Research Report RJ-9147, IBM Almaden Research Center, San José, CA, December 1992.
[2] Research supported in part by ONR contracts N00014-91-C-0026 and N00014-94-C-0007.

The LINEAR SEPARABILITY problem can be viewed as a special case of the MEMBERSHIP problem for a class C of Boolean functions: given f in DNF, decide whether f belongs to C. In this paper we investigate the complexity of the MEMBERSHIP problem for some geometrically defined classes of functions, such as linearly separable functions, functions of threshold order at most k (for fixed $k \geqslant 0$), unions/intersections of a fixed number of halfspaces, and binary-parameter analogues of these classes.

We give a general argument showing that the MEMBERSHIP problem is co-NP-hard for any class having some rather benign closure properties. This argument is then used to establish a basic co-NP-hardness result on the complexity of the MEMBERSHIP problem for all classes considered in the paper (this co-NP-hardness result has been already given for halfspaces in [19], and for functions of threshold order at most k in [24]).

In fact, in many cases we show that the considered problems are co-NP-complete. While for linearly separable functions and functions of fixed threshold order the co-NP-inclusion proofs are based on Helly's theorem and a linear programming approach, for the binary-parameter classes — zero–one threshold functions, Hamming balls and small-weight halfspaces — we make use of characterizations of these classes based on some properties of the minterms of their elements. We also obtain that the MEMBERSHIP problem for the class of unions of k halfspaces ($k \geqslant 3$ fixed) is NP-hard, co-NP-hard and belongs to Σ_2^p, hence it is not contained in NP$\cup$co-NP unless NP $=$ co-NP (for the case $k = 2$ we show only co-NP-hardness and inclusion in Σ_2^p). Thus one seems to get higher in the polynomial hierarchy when considering the complexity of the MEMBERSHIP problem for unions of $k \geqslant 3$ halfspaces. Finally, we prove that the threshold number of Boolean function given in DNF cannot even be efficiently approximated unless P $=$ NP, i.e., it is NP-hard to find a representation of a Boolean function as a union of halfspaces such that the number of elements of this decomposition is bounded above by *any* fixed function of the optimal size.

The hardness results themselves do not seem to be surprising, as MEMBERSHIP-type problems are usually not expected to be solvable in deterministic polynomial time. However, there is a motivation for studying the *exact* complexity of MEMBERSHIP problems (i.e., their location in the polynomial-time hierarchy) coming from computational learning theory (see [22, 2] for an introduction to the field), as some recent results [1, 10, 11] show connections between the complexity of the MEMBERSHIP problem and learnability issues. In fact, all our co-NP-inclusion results could be proved using a non-deterministic simulation of the known on-line learning algorithms [16, 17, 12] for the considered classes, as described in [1, 10, 11] (see [11] for results along this line), while the NP-hardness result on the complexity of the MEMBERSHIP problem for the class of unions of $k \geqslant 3$ halfspaces implies negative results on the trainability of simple neural networks in some on-line learning models [12]. This motivation makes the MEMBERSHIP problem for unions of 2 halfspaces even more interesting: showing that it is NP-hard would lead to applications in computational learning theory.

2. Definitions and notation

For $n \geqslant 1$, denote $X_n = \{0, 1\}^n$, $F_n = \{f \mid f: X_n \to \{0, 1\}\}$. Alternatively, a Boolean function $f \in F_n$ is a function of zero–one valued variables $x_1, \ldots, x_n$. A *literal* z_i is either the variable x_i or its negation $\bar{x}_i$. A *term* is a conjunction of literals, and a *clause* is a disjunction of literals. We say that a function $f \in F_n$ is given in *disjunctive normal form* (*DNF*) if it is represented as a disjunction of terms, and that it is in *k-DNF* if it is represented in a DNF with at most k literals per term; the *CNF* and *k-CNF* representations are defined analogously as conjunctions of clauses.

A function $f \in F_n$ is *non-decreasing* (*non-increasing*) *in* x_i if for every $\boldsymbol{y}, \boldsymbol{z} \in X_n$ such that $y_i = 0$, $z_i = 1$, and $y_i = z_j$ for $j \neq i$, $f(\boldsymbol{y}) \leqslant f(\boldsymbol{z})$ ($f(\boldsymbol{y}) \geqslant f(\boldsymbol{z})$) holds. A function $f \in F_n$ is *unate in variable* x_i if it is either non-decreasing or non-increasing in x_i. A function $f \in F_n$ is called *monotone* if it is non-decreasing in all its variables, and it is called *unate* if it is unate in all its variables. The term *monotone DNF* means that the given DNF representation contains no negated variables, and the term *unate DNF* means that it does not contain both a variable and its negation.

A set of literals S is a *minterm* of $f \in F_n$ if for every vector $\boldsymbol{x} \in X_n$ that assigns 1 to every literal in S we have $f(\boldsymbol{x}) = 1$, and this property holds for no proper subset S' of S.

Throughout the paper, the term NP-hard (co-NP-hard) means NP-hard (co-NP-hard) under $\leqslant_m^p$ reductions (polynomial time many–one reductions) (for details on $\leqslant_m^p$ reductions and other notions from complexity theory used in the paper, see [7, 3, 14].

3. A general argument

A *class of Boolean functions* is a sequence $C = \{C_n\}_{n \geqslant 1}$, $C_n \subseteq F_n$. In some cases we will also assume that every class contains the two 0-ary (zero–one valued) constant functions. For a class C and a Boolean function f we use the notation $f \in C$ as a shorthand for "$f \in C_l$, where l is the arity of f".

Given a class of functions $C = \{C_n\}_{n \geqslant 1}$, consider the following decision problem.
MEMBERSHIP FOR C
Instance: A Boolean function f in DNF.
Question: Is $f \in C$?

Generalizing a reduction from [19], we will show that the MEMBERSHIP problem is co-NP-hard for any class having some rather benign closure properties.

A class C is *closed under projection* iff for any $n \geqslant 1$ and any function $f \in C_n$, fixing some variables of f to 0 or 1 produces a function still in C.

We say that a class C has the *projection property* iff

(i) C is closed under projection;

(ii) for every $n \geqslant 1$, the n-ary constant *one* function belongs to C_n;

(iii) there exists an integer l such that $F_l \backslash C_l \neq \emptyset$.

Theorem 3.1. *If C is any class of Boolean functions having the projection property, then the* MEMBERSHIP *problem for C is* co-NP-*hard.*

Proof. Let C be a class having the projection property. We reduce the DNF TAUTOLOGY problem to MEMBERSHIP for C.
DNF TAUTOLOGY
Instance: A Boolean function f in DNF.
Question: Is $f \equiv 1$?
DNF TAUTOLOGY is co-NP-complete, as its complementary problem — DNF NON-TAUTOLOGY — is known to be NP-complete [7].

The reduction from DNF TAUTOLOGY to MEMBERSHIP for C is as follows. Let $l \geqslant 1$ be an integer such that $F_l \backslash C_l \neq \emptyset$ (guaranteed by point (iii) of the definition), and let $h(\boldsymbol{y}) \in F_l \backslash C_l$ be in DNF. Given an instance of DNF TAUTOLOGY, i.e., a function $f(\boldsymbol{x}) \in F_n$ in DNF, construct the DNF for the function

$$g(\boldsymbol{x},\boldsymbol{y}) = f(\boldsymbol{x}) \vee h(\boldsymbol{y}) \in F_{n+l}$$

(as h is fixed, this can be done in time polynomial in the length of the description of f). If $f \equiv 1$, then $g(\boldsymbol{x},\boldsymbol{y}) \equiv 1$, thus — by point (ii) of the definition — $g(\boldsymbol{x},\boldsymbol{y}) \in C_{n+l}$. If not, let $\boldsymbol{a} \in X_n$ be such that $f(\boldsymbol{a}) = 0$. Then $g(\boldsymbol{a},\boldsymbol{y}) = h(\boldsymbol{y}) \notin C_l$, and — as C is closed under projection — also $g(\boldsymbol{x},\boldsymbol{y}) \notin C_{n+l}$. That is, $f \equiv 1$ if and only if $g \in C_{n+l}$, which completes the proof. $\square$

Notice that — as DNF TAUTOLOGY remains co-NP-complete even when restricted to instances in 3-DNF [7] — if we have a function not in C that can be represented in k-DNF, then the above argument gives that the MEMBERSHIP problem for C remains co-NP-hard even when restricted to instances in $\max\{k,3\}$-DNF.

4. Hardness results

Theorem 3.1 can be readily applied to show that, e.g., the MEMBERSHIP problem for the class monotone, unate and symmetric functions is co-NP-complete, even when restricted to instances in 3-DNF (the inclusions in co-NP are clear). However, here we are interested in the complexity of the MEMBERSHIP problem for some geometrically defined classes of functions.

For a function $f \in F_n$, denote

$$\mathrm{POS}(f) = \{\boldsymbol{x} \mid \boldsymbol{x} \in X_n \text{ and } f(\boldsymbol{x}) = 1\}.$$

A Boolean function $f \in F_n$ is *linearly separable* (we will also use the term that f is a *halfspace*) iff there exist $w_1, \ldots, w_n$, $t \in \Re$ such that

$$\mathrm{POS}(f) = \left\{\boldsymbol{x} \mid \boldsymbol{x} \in X_n \text{ and } \sum_{i=1}^{n} w_i x_i \geqslant t\right\}.$$

The MEMBERSHIP problem for the class of linearly separable functions, i.e., the LINEAR SEPARABILITY problem mentioned in the introduction, is known to be solvable in deterministic polynomial time when restricted to instances in monotone DNF [19]. However, as the class of linearly separable functions clearly has the projection property, Theorem 3.1 implies that the general LINEAR SEPARABILITY problem is co-NP-hard (this is essentially the result given in [19]). We now prove that it is in fact co-NP-complete. The co-NP inclusion proof is based on a classical result in combinatorial geometry — Helly's theorem[3] [13,6] (in fact, a weak version of the theorem suffices here).

Theorem 4.1 (Helly [13,6]). *For any finite collection $S_1, \dots, S_k$ of $k \geqslant d+1$ distinct convex sets in $\mathfrak{R}^d$,*

$$\bigcap_{i=1}^{k} S_i \neq \emptyset \quad \textit{iff there exist } 1 \leqslant i_1 < \cdots < i_{d+1} \leqslant k \textit{ such that } S_{i_1} \cap \cdots \cap S_{i_{d+1}} = \emptyset.$$

Theorem 4.2. LINEAR SEPARABILITY *is* co-NP-*complete, even when restricted to instances in* 3-*DNF.*

Proof. The hardness result is implied by Theorem 3.1 and the fact that the function $y_1 y_2 \vee \bar{y}_1 \bar{y}_2 \in F_2$ is not a halfspace (see the note after Theorem 3.1).

It remains to show that LINEAR SEPARABILITY is in co-NP. First, one can see that the question whether a Boolean function f is linearly separable is equivalent to the question whether a certain system of 2^n linear inequalities with $n+1$ unknowns (the coefficients are formed by the elements $e \in X_n$, the unknowns are $w_1, \dots, w_n, t$, and the value of f at a point $e = (e_1, \dots, e_n) \in X_n$ determines whether the inequality $\sum_{i=1}^n e_i w_i - t \geqslant 0$ or $\sum_{i=1}^n e_i w_i - t < 0$ is considered) has any solution, i.e., whether the given (open or closed) halfspaces in $\mathfrak{R}^{n+1}$ have non-empty intersection. Helly's theorem then says that the system has *no* solution if and only if it contains a "small" subsystem that has no solution either, implying that the following non-deterministic algorithm can be used to decide whether a given function $f \in F_n$ is *not* linearly separable.

Algorithm 4.3.

(0) Input: A Boolean function $f \in F_n$ in DNF.
(1) Select, non-deterministically, $n+2$ distinct points from X_n.
(2) Construct the system of linear inequalities corresponding to f and the selected points.
(3) Solve the obtained LP problem.[4]
(4) If the LP problem is infeasible, then ACCEPT.

[3] One could also use Caratheodory's theorem as done in [5] for a similar problem.

[4] Because of its dimensions $(n+2) \times (n+1)$, this problem is trivial to solve in polynomial time.

As the coefficients in the LP problem are from the set $\{-1,0,1\}$, the algorithm clearly runs in time polynomial in the length of the description of f, and accepts f if and only if f is not linearly separable. □

A negation argument gives that the version of LINEAR SEPARABILITY with instances in CNF is also co-NP-complete.

A natural way to generalize the above result is to consider Boolean functions of higher threshold order (defined in [24]). We say that a pair $(w;t)$, $w \in \Re^{\ell}$ ($\ell = \sum_{i=1}^{k}\binom{n}{i}$, $k \in N$), $t \in \Re$, is a *separator of degree k* for some $f \in F_n$ if and only if

$$\mathrm{POS}(f) = \left\{ x \mid x \in X_n \text{ and } \sum_{\substack{S \subset \{1,\dots,n\} \\ 1 < |S| \leqslant k}} w_S \prod_{i \in S} x_i \geqslant t \right\}$$

(assume that the indexing is actually made according to some fixed ordering of the non-empty subsets of $\{1,\dots,n\}$). A function $f \in F_n$ is called a *threshold function of order k* if and only if it has a separator of degree k, but no separator of degree $k-1$. For any dimension n, the two constant functions are threshold functions of order 0. Clearly, the class of linearly separable functions consists of all functions of order 0 and 1.

For any fixed $k \geqslant 0$, let k-THRESHOLD ORDER RECOGNITION be the MEMBERSHIP problem for the class of Boolean functions of threshold order at most k.

Theorem 4.4. *For any fixed $k \geqslant 0$, the k-*THRESHOLD ORDER RECOGNITION *problem is* co-NP-*complete, even when restricted to instances in* $\max\{k+1,3\}$-*DNF.*

Proof. As the class of functions of threshold order at most k has the projection property, the co-NP-hardness result follows from Theorem 3.1 (this is essentially the result given in [24]). The restriction to $\max\{k+1,3\}$-DNF is implied by the fact that the $k+1$-ary parity function is of threshold order $k+1$ [24].

To show that k-THRESHOLD ORDER RECOGNITION is in co-NP, a generalization of the above approach for halfspaces, i.e., for the case $k = 1$, can be used. Once again, we have a system of 2^n linear inequalities with $1 + \sum_{i=1}^{k}\binom{n}{i}$ unknowns (i.e., polynomially many in n for any fixed k), and the coefficients are from $\{-1,0,1\}$, so it suffices to guess $2 + \sum_{i=1}^{k}\binom{n}{i}$ distinct inequalities and accept if and only if the obtained LP problem is infeasible. □

Once again, a negation argument gives that the version of k-THRESHOLD ORDER RECOGNITION with instances in CNF is also co-NP-complete, as it is known that f is of threshold order k if and only if its negation is of threshold order k [24].

A further natural geometrically defined class of functions could be the class of balls. However, it turns out that in our setting this class is identical to the class of halfspaces (we observe this identity because of the analogous binary classes considered later).

A Boolean function $f \in F_n$ is a *ball* if and only if there exist $w_1, \dots, w_n, r \in \Re$, $r \geqslant 0$, such that $\mathrm{POS}(f) = \{x \mid x \in X_n \text{ and } \sum_{i=1}^{n} (w_i - x_i)^2 \leqslant r^2\}$.

Lemma 4.5. *A Boolean function is a ball if and only if it is linearly separable.*

Proof. Let $f \in F_n$ be linearly separable, i.e.,

$$\mathrm{POS}(f) = \left\{ x \mid x \in X_n \text{ and } \sum_{i=1}^{n} w_i x_i \geqslant t \right\}$$

for some $w_1, \dots, w_n, t \in \Re$. Then one can check that

$$\mathrm{POS}(f) = \left\{ x \mid x \in X_n \text{ and } \sum_{i=1}^{n} \left(\frac{1 + w_i}{2} - x_i \right)^2 \leqslant -t + \frac{1}{4} \sum_{i=1}^{n} (w_i + 1)^2 \right\},$$

i.e., f is a ball.

Conversely, if $f \in F_n$ is a ball, i.e.,

$$\mathrm{POS}(f) = \left\{ x \mid x \in X_n \text{ and } \sum_{i=1}^{n} (v_i - x_i)^2 \leqslant r^2 \right\}$$

for some $v_1, \dots, v_n$ $r \in \Re$, $r \geqslant 0$, then it follows that

$$\mathrm{POS}(f) = \left\{ x \mid x \in X_n \text{ and } \sum_{i=1}^{n} (2v_i - 1)x_i \geqslant \sum_{i=1}^{n} v_i^2 - r^2 \right\},$$

giving that f is linearly separable. □

So far, the co-NP inclusion results have been proved using Helly's theorem and the existence of polynomial-time algorithms for linear programming. For more restricted geometrical concept classes considered in the sequel, these tools cannot be used any more. It would be still possible to use a non-deterministic simulation of the known efficient on-line learning algorithms for such classes as described in [1, 10, 11], but here we give more direct inclusion proofs.

A Boolean function $f \in F_n$ is a *zero–one threshold function* [12] if and only if there exist $w_1, \dots, w_n \in [0, 1]$, $t \in \mathcal{N}$, such that $\mathrm{POS}(f) = \{x \mid x \in X_n \text{ and } \sum_{i=1}^{n} w_i x_i \geqslant t\}$ (functions of this type are called Boolean threshold functions in [20]: we have changed the terminology to avoid confusion). Observe that $f \in F_n$ is a non-constant zero–one threshold function such that $\mathrm{POS}(f) = \{x \mid x \in X_n \text{ and } \sum_{i=1}^{n} w_i x_i \geqslant t\}$ if and only if the minterms of f are all t-element subsets of the set of variables $\{x_i \mid w_i = 1 \text{ and } 1 \leqslant i \leqslant n\}$. Using this fact, one can decide whether f is a zero–one threshold function by computing the minterms of f and then checking whether they define a zero–one threshold function. If f is in *monotone* DNF, then the whole computation can be done in time polynomial in the length of the description of f (see any textbook on switching theory for details).

Let ZERO–ONE THF RECOGNITION be the MEMBERSHIP problem for the class of zero–one threshold functions.

Theorem 4.6. ZERO–ONE THF RECOGNITION is co-NP-*complete, even when restricted to instances in* 3-*DNF.*

Proof. The co-NP-hardness result follows from Theorem 3.1 and the fact that the function $y_1y_2 \vee \bar{y}_1\bar{y}_2 \in F_2$ is not a zero–one threshold function.

To prove the inclusion in co-NP, consider the following non-deterministic algorithm.

Algorithm 4.7.

```
(0) Input: A Boolean function $f \in F_n$ in DNF.
(1) Delete the negated variables from the DNF representation of $f$. Denote $g \in F_n$
    the function represented by the modified (monotone) DNF formula.
(2) Compute the minterms of $g$, and check whether they define a zero–one threshold
    function. If yes, let $h = g$; if not, let $h \in F_n$ be the constant one function.
(3) Select, non-deterministically, a point $x \in X_n$.
(4) If $f(x) \neq h(x)$ then ACCEPT.
```

As the function g constructed in step (1) is in monotone DNF, its minterms can be computed in time polynomial in the length of the description of g (as observed above). This implies that Algorithm 4.7 runs in time polynomial in the length of the description of f.

To prove the correctness of the algorithm, notice that if f is a monotone function, then $g \equiv f$ in step (1). That is, if f *is* a zero–one threshold function, then the function h constructed in step (2) is logically equivalent to f. Thus, if f *is* a zero–one threshold function, we surely do not accept in step (4). On the other hand, if f is *not* a zero–one threshold function, then — as the function h constructed in step (2) always *is* a zero–one threshold function — it must be the case that $h \not\equiv f$, and a computation accepting f in step (4) must exist. That is, f is accepted if and only if it is *not* a zero–one threshold function, which completes the proof. □

Let us continue with a binary version of balls. A Boolean function $f \in F_n$ is called a *Hamming ball* [12] if and only if there exist $w_1, \ldots, w_n \in [0,1]$, $k \in \mathscr{Z}$, such that $\mathrm{POS}(f) = \{x \mid x \in X_n \text{ and } \sum_{i=1}^{n} |w_i - x_i| \leqslant k\}$. It can be shown that Hamming balls are exactly those halfspaces which are representable using weights from the set $\{-1, 1\}$ [12]. Observe that a non-constant function $f \in F_n$ is a Hamming ball such that $\mathrm{POS}(f) = \{x \mid x \in X_n \text{ and } \sum_{i=1}^{n} |w_i - x_i| \leqslant k\}$ if and only if the minterms of f are all $n-k$-element subsets of the set of literals $\{z_i \mid z_i = x_i \text{ if } w_i = 1 \text{ and } z_i = \bar{x}_i \text{ otherwise}, 1 \leqslant i \leqslant n\}$. One can thus decide whether f is a Hamming ball by computing its minterms and checking whether they define a Hamming ball. If f is given in *unate* DNF, then all this can be done in time polynomial in the length of the description of f (argue similarly as in the monotone case).

Let HAMMING BALL RECOGNITION be the MEMBERSHIP problem for the class of Hamming balls.

Theorem 4.8. HAMMING BALL RECOGNITION *is* co-NP-*complete, even when restricted to instances in* 3-*DNF*.

Proof. Again, the co-NP-hardness result follows from Theorem 3.1 and the fact that the function $y_1 y_2 \vee \bar{y}_1 \bar{y}_2 \in F_2$ is not a Hamming ball. □

To prove the inclusion in co-NP, consider the following non-deterministic algorithm.

Algorithm 4.9.

```
(0) input: A Boolean function $f \in F_n$ in DNF.
(1) Select, non-deterministically, $\boldsymbol{u}, \boldsymbol{v} \in X_n$. Continue if $u_i = 0$, $v_i = 1$, $u_j = v_j$ for
    $j \neq i$. (for some $i$), and $f(\boldsymbol{u}) \neq f(\boldsymbol{v})$; otherwise stop.
(2) Assume w.l.o.g. that $f(\boldsymbol{u}) = 1$ and $f(\boldsymbol{v}) = 0$ (the other case is analogous).
(3) For $i = 1, 2, \dots, n$ do
        if $f(u_1, \dots, u_{i-1}, 1 - u_i, u_{i+1}, \dots, u_n) = 0$ then let $w_i = u_i$
        else let $w_i = 1 - u_i$.
(4) For $i = 1, 2, \dots, n$ do
        if $w_i = 1$ then let $z_i = \bar{x}_i$ else let $z_i = x_i$.
(5) Delete the literals $z_i$ from the DNF representation of $f$ (for $i = 1, 2, \dots, n$).
    Denote $g \in F_n$ the function represented by the modified (unate) DNF formula.
(6) Compute the minterms of $g$, and check whether they define a Hamming ball. If
    yes, let $h = g$; if not, let $h \in F_n$ be the constant one function.
(7) Select, non-deterministically, a point $\boldsymbol{x} \in X_n$.
(8) If $f(\boldsymbol{x}) \neq h(\boldsymbol{x})$ then ACCEPT.
```

As the function g constructed in step (4) is in unate DNF, its minterms can be computed in time polynomial in the length of the description of g. This implies that Algorithm 4.9 runs in time polynomial in the length of the description of f.

To prove the correctness of the algorithm, notice that if f *is* a Hamming ball, then — as the center of f in step (3) and the "signs" of the variables of f in step (4) are computed correctly, and then $g \equiv f$ — the function h constructed in step (6) is logically equivalent to f. Thus, if f *is* a Hamming ball, we surely do not accept in step (8). On the other hand, if f is *not* a Hamming ball, then — as the function h constructed in step (6) always *is* a Hamming ball — it is the case that $h \not\equiv f$, and a computation accepting f in step (8) must exist. That is, f is accepted if and only if it is *not* a Hamming ball, which completes the proof. □

A negation argument gives that the version of HAMMING BALL RECOGNITION with instances in CNF is also co-NP-complete.

A Boolean function $f \in F_n$ is called a *small-weight halfspace* [12] if and only if there exist $w_1, \dots, w_n \in \{-1, 0, 1\}$, $t \in \mathscr{Z}$, such that $\mathrm{POS}(f) = \{\boldsymbol{x} \mid \boldsymbol{x} \in X_n \text{ and } \sum_{i=1}^{n} w_i x_i \geq t\}$. Let SMALL-WEIGHT HSP RECOGNITION be the MEMBERSHIP problem for the class of small-weight halfspaces.

Theorem 4.10. *The* SMALL-WEIGHT HSP RECOGNITION *problem is* co-NP-*complete, even when restricted to instances in* 3-*DNF.*

Proof. The proof is analogous to that for Hamming balls (the case of irrelevant variables has to be handled). □

Again, the version of SMALL-WEIGHT HSP RECOGNITION with instances in unate DNF is solvable in deterministic polynomial time.

We say that a Boolean function $f \in F_n$ is a *union of k halfspaces* if f can be expressed as a disjunction of k (not necessarily distinct) linearly separable functions from F_n. A *threshold decomposition* of a Boolean function f is a collection of linearly separable functions whose disjunction is equivalent to f. The *threshold number* $t(f)$ of f is defined as the smallest integer k such that there is a k-element threshold decomposition for f (we do not allow empty disjunctions, hence $t(f) \geqslant 1$ for any Boolean function f). An *optimal threshold decomposition* of a Boolean function f is a threshold decomposition of f with $t(f)$ elements.

We are interested in the following decision problem (for fixed $k \geqslant 1$).

UNION OF k HALFSPACES

Instance: A Boolean function f in DNF.

Question: Is the threshold number of f at most k?

First consider a restricted version of this problem. Denote by $\leqslant$ the common partial order on X_n, i.e., for $\boldsymbol{x},\boldsymbol{y} \in X_n, \boldsymbol{x} \leqslant \boldsymbol{y}$ iff $x_i \leqslant y_i$ for all i $(1 \leqslant i \leqslant n)$. A monotone Boolean function is uniquely determined by its *minimal* 1's, by the vectors $\boldsymbol{x}$ such that $f(\boldsymbol{x}) = 1$ and for every $\boldsymbol{y} \neq \boldsymbol{x}$ such that $\boldsymbol{y} \leqslant \boldsymbol{x}$, necessarily $f(\boldsymbol{y}) = 0$. The *norm* of a vector $\boldsymbol{x} \in X_n$ is defined as $\|\boldsymbol{x}\| = \sum_{i=1}^{n} x_i$. A monotone Boolean function f is called *graphic* if all the minimal 1's of f are of norm at most 2. Graphic functions are easily seen to be exactly those functions that are representable in monotone 2-DNF. We say that a Boolean function is a *graphic halfspace* if it is a graphic function and a halfspace.

It is known that LINEAR SEPARABILITY with instances in monotone 2-DNF is solvable in deterministic polynomial time [1], and the same holds true for the UNION OF 2 HALFSPACES problem as well [15]. However, for $k \geqslant 3$ the problem becomes harder.

Theorem 4.11 [25, 23]. *For any fixed* $k \geqslant 3$, UNION OF k HALFSPACES *with instances in monotone* 2-*DNF is* NP-*complete.*

Proof. The NP-hardness result was proved in [25] for $k = 3$; the generalization for $k \geqslant 4$ — due to [23] — follows from the case $k = 3$, as the threshold number of some $f(x_1, \ldots, x_n) \in F_n$ in monotone 2-DNF is at most 3 if and only if the threshold number of $f(x_1, \ldots, x_n) \vee y_1y_2 \vee y_3y_4 \vee \cdots \vee y_{2(k-3)-1}y_{2(k-3)} \in F_{n+2(k-3)}$ is at most k.

In fact, the hardness results in [25,23] are on the problem of deciding whether graphic function is a union of k *graphic* halfspaces. However, they are also valid in the form given here, as it is known that a graphic function is a union of l halfspaces if and only if it is a union of l *graphic* halfspaces [23]. We sketch the proof of this fact for completeness.

For a monotone Boolean function $h \in F_n$ denote M_h the set of its minimal 1's, and let $h^{(\mathrm{gr})} \in F_n$ be the monotone function such that $M_{h^{(\mathrm{gr})}} = \{\boldsymbol{x} \mid \boldsymbol{x} \in M_h \text{ and } \|\boldsymbol{x}\| \leqslant 2\}$. It is known that if a monotone function h is a halfspace, then also $h^{(\mathrm{gr})}$ is a (graphic) halfspace (this follows from Theorem 10.4 in [8]; another proof is given in [23]).

Now assume that a graphic function f is a union of l halfspaces, i.e., let $f = f_1 \vee \cdots \vee f_l$, where the f_i's are halfspaces. As f is a monotone function, without loss of generality the f_i's can be also assumed to be monotone [9]. Then we have $f = f_1^{(\mathrm{gr})} \vee \cdots \vee f_l^{(\mathrm{gr})}$, where $f_1^{(\mathrm{gr})}, \dots, f_l^{(\mathrm{gr})}$ are graphic halfspaces, which completes the proof of claim.

It remains to show that for any fixed $k \geqslant 3$ the UNION OF k HALFSPACES problem with instances in monotone 2-DNF is in NP.

Using Theorem 10.4 from [8] and some standard bounds on the size of weights required to represent halfspaces [18, 17], it can be shown that a graphic function $f \in F_n$ is a (graphic) halfspace if and only if there exist non-negative integers $w_1, \dots, w_n$, t having $O(n \log n)$ bits long binary encodings such that for all $\boldsymbol{x} \in X_n$, $\|\boldsymbol{x}\| \leqslant 2$, $f(\boldsymbol{x}) = 1$ if and only if $\sum_{i=1}^{n} w_i x_i \geqslant t$. Then — as we have seen that any threshold decomposition of a graphic function can be assumed to consist of graphic halfspaces — the following non-deterministic algorithm decides in polynomial time whether a graphic function $f \in F_n$ is a union of k halfspaces. Select, non-deterministically, some $O(n \log n)$ bits long integer coefficients defining k graphic halfspaces $f_1, \dots, f_k \in F_n$ (in the sense of the above representation scheme), and accept if and only if $f = f_1 \vee \cdots \vee f_k$. Clearly, as both f and $f_1 \vee \cdots \vee f_k$ are graphic functions, they are identical if and only if $M_f = M_{f_1 \vee \cdots \vee f_k}$, what can be decided in deterministic polynomial time evaluating f and $f_1 \vee \cdots \vee f_k$ on the elements of the set $\{\boldsymbol{x} \mid \boldsymbol{x} \in X_n$ and $\|\boldsymbol{x}\| \leqslant 2\}$. □

The general UNION OF k HALFSPACES problem is clearly not easier than its restriction to instances in monotone 2-DNF.

Theorem 4.12. (i) UNION OF 2 HALFSPACES *is* co-NP-*hard and belongs to* Σ_2^p.

(ii) *For any fixed* $k \geqslant 3$, UNION OF k HALFSPACES *is* NP-*hard*, co-NP-*hard*, *and belongs to* Σ_2^p.

Proof. The co-NP-hardness result follows in both cases from Theorem 3.1, as the considered classes have the projection property. The NP-hardness result for part (ii) follows from Theorem 4.11. Using the fact that for any $l \geqslant 1$, the threshold number of the function $y_1 y_2 \vee y_3 y_4 \vee \cdots y_{2l-1} y_{2l} \in F_{2l}$ is exactly l [6, 9], all hardness results can be shown to remain valid even when the problems are restricted to instances in 3-DNF. The inclusions in Σ_2^p are proved in all cases by the algorithm that guesses, existentially, a threshold decomposition, and then verifies, universally, whether the decomposition is logically equivalent to the instance of the decision problem (all this computation can be done efficiently, as it is known [18, 17] that any halfspace $f \in F_n$ can be represented using integer weights with $O(n \log n)$ bits long binary encoding). $\square$

Corollary 4.13. *For any fixed* $k \geqslant 3$, *the* UNION OF k HALFSPACES *problem is not in* NP$\cup$co-NP *unless* NP = co-NP.

Proof. As both NP and co-NP are closed under $\leqslant_m^p$ reductions, no NP-hard problem can be in co-NP unless NP = co-NP, and no co-NP-hard problem can be in NP unless NP = co-NP. $\square$

We now turn out attention to the following optimization problem (considered in [9]).
OPTIMAL THRESHOLD DECOMPOSITION
Instance: A Boolean function f in DNF.
Task: Find an optimal threshold decomposition for f.
This problem is related to the decision problem
THRESHOLD NUMBER
Instance: A Boolean function f in DNF and an integer K.
Question: Is the threshold number of f at most K?
which is not easier than its previous fixed-parameter analogues.

Theorem 4.14. THRESHOLD NUMBER *is* NP-*hard*, co-NP-*hard*, *and belongs to* Σ_2^p.

Proof. Follows from our previous arguments (in fact, the NP-hardness result was given already in [6]). $\square$

Corollary 4.15. THRESHOLD NUMBER *is not in* NP$\cup$co-NP *unless* NP = co-NP.

Corollary 4.16 (Danzer et al. [6]). OPTIMAL THRESHOLD DECOMPOSITION *is* NP-*hard.*

Moreover, using the fixed-parameter hardness results from Theorem 4.12 and standard arguments from [7], one can show that the optimal threshold decomposition cannot be efficiently approximated (with respect to the number of elements) within any constant factor $\rho < \frac{3}{2}$ unless P = NP. However, we will show that this problem is not efficiently approximable in a much stronger sense (the definition is motivated by similar notions in [21] used for a somewhat different problem).

Definition 4.17. Let Π be a minimization problem, and h be any function of the single variable opt. We say that Π is *h(opt)-approximable* if and only if there exists a constant

α and a polynomial time algorithm APPROX such that on input of any instance I of Π for which $\mathrm{opt}(I) \geqslant \alpha$, APPROX outputs a feasible solution with cost at most $h(\mathrm{opt}(I))$ (notice that the algorithm is required to perform well only asymptotically, from some α on).

Theorem 4.18. *The* OPTIMAL THRESHOLD DECOMPOSITION *problem is not* $h(\mathrm{opt})$*-approximable for any function h unless* P = NP.

Proof. Let $\alpha \geqslant 1$ be any fixed higher, h any function of a single variable, and k any integer such that $k \geqslant h(\alpha) + 1$ and $k \geqslant \alpha + 1$. Consider the following reduction from DNF TAUTOLOGY to UNION OF α HALFSPACES. Given $f(\boldsymbol{x}) \in F_n$ in DNF, construct the DNF for the function

$$g(\boldsymbol{x}, y_1, y_2, \ldots, y_{2\alpha}, v_1, v_2, \ldots, v_k, w_1, w_2, \ldots, w_k)$$
$$= f(\boldsymbol{x}) y_1 y_2 \vee \left(\bigvee_{i=1}^{k} y_1 y_2 v_i w_i \right) \vee y_3 y_4 \vee \cdots \vee y_{2\alpha-1} y_{2\alpha}.$$

If $f \equiv 1$ then the threshold number of g is α (use the fact that the threshold number of the function $y_1 y_2 \vee y_3 y_4 \vee \cdots y_{2l-1} y_{2l} \in F_{2l}$ is exactly l). If $f \not\equiv 1$, let $\boldsymbol{a} \in X_n$ be such that $f(\boldsymbol{a}) = 0$. Then the projection $g(\boldsymbol{a}, y_1, y_2, \ldots, y_{2\alpha}, v_1, v_2, \ldots, v_k, w_1, w_2, \ldots, w_k)$ has threshold number at least k, hence g has threshold number at least $k \geqslant \alpha + 1$. That is, $f \equiv 1$ if and only if the threshold number of g is at most α, thus UNION OF α HALFSPACES remains co-NP-hard for instances of the constructed type. As the threshold number of the constructed instances is either α or at least $h(\alpha) + 1$, an $h(\mathrm{opt})$-approximation algorithm which performs well on all instances with threshold number at least α would solve the above co-NP-hard problem in polynomial time. As h and α were arbitrary, this concludes the proof. $\square$

Notice that using a negation argument all results on the complexity of the UNION of k HALFSPACES problem can be shown to hold also for the dual INTERSECTION OF k HALFSPACES problem with instances in CNF. One can also see that our non-approximately result on threshold decompositions still holds if the approximation algorithm only *decides* whether there exists a threshold decomposition with fewer elements than the upper bound.

Acknowledgements

The authors thank the two anonymous referees for observing the result given in Lemma 4.5 and the co-NP-inclusion part of Theorem 4.6, for pointing out the references [5, 15], and for a large number of other constructive comments.

References

[1] H. Aizenstein, L. Hellerstein and L. Pitt, Read-thrice DNF is hard to learn with membership and equivalence queries, in: Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'92) (IEEE Computer Society Press, Silver Spring, MD, 1992) 523–532.

[2] D. Angluin, Computational learning theory: survey and selected bibliography, in: Proceedings of the 24th Annual ACM Symposium on the Theory of Computing (STOC'92) (ACM Press, New York, 1992) 351–369.

[3] J.L. Balcàzar, J. Diaz and J. Gabarró, Structural Complexity I (Springer, Berlin, 1988).

[4] V. Chvátal and P.L. Hammer, Aggregation of linear inequalities in integer programming, Ann. Discrete Math. 1 (1977) 145–162.

[5] Y. Crama, Recognition problems for special classes of polynomials in 0–1 variables, Math. Programming 44 (1989) 139–155.

[6] L.W. Danzer, B. Grünbaum and V. Klee, Helly's theorem and its relatives, in: Convexity, Proc of Symposia in Pure Mathematics, Vol. 7 (AMS, Providence, 1963) 101–180.

[7] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness (Freeman, San Francisco, 1979).

[8] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs (Academic Press, New York, 1980).

[9] P.L. Hammer, T. Ibaraki and U.N. Peled, Threshold numbers and threshold completions, in: P. Hansen, ed., Studies on Graphs and Discrete Mathematics (North-Holland, Amsterdam, 1981) 125–145.

[10] T. Hegedűs, Computational limitations on PAC and on-line learning over the Boolean domain: a comparison, unpublished manuscript, 1992.

[11] T. Hegedűs, Can complexity theory benefit from learning theory?, in: Proceedings of the European Conference on Machine Learning (ECML-93), Vienna, 1993 (Lecture Notes in Computer Science, Vol. 667, Springer, Berlin, 1993) 354–359.

[12] T. Hegedűs, On training simple neural networks and small-weight neurons, in: J. Shawe-Taylor and M. Anthony, eds., Computational Learning Theory: EuroCOLT'93 (Clarendon Press, Oxford, 1994) 69–82.

[13] E. Helly, Über Mengen konvexer Körper mit gemeinschaftlichen Punkten, Jahresbericht der Deutschen Mathematiker-Vereinigung 32 (1923) 175–176.

[14] D.S. Johnson, A catalog of complexity classes, in: J. van Leeuwen, ed., Handbook of Theoretical Computer Science, Vol. A (Elsevier, Amsterdam, 1990), 67–161.

[15] T.H. Ma, On the threshold dimension 2 graphs, Technical Report, Institute of Information Science, Academia Sinica, Nankang, Taipei, Republic of China, 1993.

[16] W. Maass and Gy. Turán, On the complexity of learning from counterexamples, in: Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science (FOCS'89) (IEEE Computer Society Press, Los Angeles, 1989) 262–267.

[17] W. Maass and Gy. Turán, How fast can a threshold gate learn?, Report 321, IIG-Report Series, Graz University of Technology, 1991.

[18] S. Muroga, Threshold Logic (Wiley/Interscience, New York, 1971).

[19] U.N. Peled and B. Simeone, Polynomial-time algorithms for regular set-covering and threshold synthesis, Discrete Appl. Math. 12 (1985) 57–69.

[20] L. Pitt and L. Valiant, Computational limitations on learning from examples, J. ACM 35 (1988) 965–984.

[21] L. Pitt and M.K. Warmuth, The minimum consistent DFA problem cannot be approximated within any polynomial, J. ACM 40(1) (1993) 95–142.

[22] Gy. Turán, A survey of some aspects of computational learning theory, in: Proceedings of the 8th Conference Fundamentals of Computation Theory (FCT'91), Szeged, Hungary, 1991, Lecture Notes in Computer Science, Vol. 529 (Springer, Berlin, 1991) 89–103.

[23] R.I. Tyshkevich and A.A. Chernyak, Threshold decompositions of Boolean functions and graphs, in: A.A. Markov, ed., Combinatorial-Algebraic and Probabilistic Methods of Discrete Analysis (Gorkii State University, 1989) 111–129 (in Russian).

[24] C. Wang and A.C. Williams, The threshold order of a Boolean function, Discrete Appl. Math. 31 (1991) 51–69.

[25] M. Yannakakis, The complexity of the partial order dimension problem, SIAM J. Algebraic Discrete Methods 3 (1982) 351–358.