On Learning Counting Functions With Queries

Zhixiang Chen*
Boston University

Steven Homer†
Boston University

Abstract

We investigate the problem of learning disjunctions of counting functions, generalizations of parity and modulo functions, with equivalence and membership queries. We prove that, for any prime number p, the class of disjunctions of integer-weighted counting functions with modulus p over the domain Z_q^n (or Z^n) for any given integer $q \geq 2$, is polynomial time learnable using at most $n+1$ equivalence queries. The hypotheses issued by the learner are disjunctions of at most n counting functions with weights from Z_p. In general a counting function may have a composite modulus. We prove that, for any given integer $q \geq 2$, over the domain Z_2^n, the class of read-once disjunctions of Boolean-weighted counting functions with modulus q is polynomial time learnable with only one equivalence query and $O(n^q)$ membership queries. And the class of disjunctions of $\log\log n$ Boolean-weighted counting functions with modulus q is polynomial time learnable.

1 Introduction

Recently, symmetric Boolean functions, especially parity functions and modulo functions, have received much attention in computational learning theory. It is known that the class of single parity functions (see Helmbold, Sloan and Warmuth [HSW]) and the class of single modulo functions with modulus p for any given prime number p (see, Blum, Chalasani and Jackson [BCJ]) are pac-learnable. Fisher and Simon [FS] proved that parity functions of monomials with at most k literals are pac-learnable, while given the assumption that $RP \neq NP$ parity functions of k monomials are not pac-learnable with the same type of functions as hypotheses for any fixed $k \geq 2$. Meanwhile, Blum and Singh [BS] showed that, for any constant k, Boolean functions of k monomials are pac-learnable by the more expressive hypothesis class of general DNF formulas. They also proved that, for any $k \geq 2$, for any fixed symmetric function f on k inputs, f consisting of k monomials is not pac-learnable with the same type of functions as hypothesis under the assumption that $RP \neq NP$.

In the on-line model with queries, Angluin, Hellerstein, and Karpinski [AHK] have shown that read-once Boolean functions over the basis (AND, OR, NOT) are polynomial time learnable with equivalence and membership queries. Hancock and Hellerstein [HH] extended this result for Boolean functions to a larger basis including arbitrary threshold functions and parity functions. Further, Bshouty, Hancock, and Hellerstein [BHH] showed that read-once functions over the basis of arbitrary symmetric functions are polynomial time learnable with equivalence and membership queries. However, they also proved that read-twice functions over the same basis are not, under standard cryptographic assumptions.

Our goal in this paper is to obtain further positive results for on-line learning of counting functions, which include parity and modulo functions, with equivalence and membership queries. Bshouty, Hancock and Hellerstein's negative result for read-twice Boolean functions over the basis of arbitrary symmetric functions is very strong. However, a key condition in their theorem is that they require the basis to include the three-input consensus function, i.e., a function outputs 1 if and only if all its inputs get the same value. However, for many specific symmetric functions, e.g., modulo functions, counting functions, and threshold functions, this condition does not hold, i.e., no one of those functions is equivalent to a consensus function.

We observe that a disjunction of integer-weighted count-

*Department of Computer Science, Boston University, Boston, MA 02215; zchen@cs.bu.edu. The author was supported by NSF grant CCR-9103055 and by a Boston University Presidential Graduate Fellowship.

†Department of Computer Science, Boston University, Boston, MA 02215; homer@cs.bu.edu. The author was supported by NSF grant CCR-9103055.

COLT 94 - 7/94 New Brunswick, N..J. USA
© 1994 ACM 0-89791-655-7/94/0007..$3.50

ing functions over a field Z_p for a given prime number p corresponds to a linear system over the field Z_p. We prove that (1) the class of homogeneous linear systems over an arbitrary field is polynomial time learnable with at most n equivalence queries, and (2) the class of linear systems over an arbitrary field is polynomial time learnable with at most $n+1$ equivalence queries. Here n is the number of input variables, the hypotheses issued in (1) by the learner are homogeneous linear systems of no more than n equations, and the hypotheses issued in (2) by the learner are also linear systems of no more than n equations. The first result implies that, for any prime number p, the class of disjunctions of integer-weighted modulo functions with modulus p over the field Z_p is polynomial time learnable with at most n queries, where the hypotheses issued by the learner are disjunctions of modulo functions with modulus p and weights from Z_p. The second result implies that, for any prime number p, the class of disjunctions of integer-weighted counting functions with modulus p is polynomial time learnable with at most $n+1$ equivalence queries, where the hypotheses issued by the learner are disjunctions of counting functions with modulus p and weights from Z_p. We also extend the above results to disjunctions of integer-weighted modulo functions (or in general integer-weighted counting functions) with different prime moduli.

The above results rely on the facts that Z_p is a field for any prime number p. When p is a composite number, however, this is not true. Nevertheless, we prove that, given any integer $q \geq 2$, the class of read-once disjunctions of Boolean-weighted counting functions over the domain Z_2^n is polynomial time learnable with only *one* equivalence query and $O(n^q)$ membership queries, where n is the number of input variables. This result cannot be subsumed by Bshouty, Hancock and Hellerstein's result [BHH] on learning read-once functions over the basis of arbitrary symmetric functions in the sense of the equivalence query complexity, since their result requires at most n^3 equivalence queries. In general, based on analyzing the "modulo-structure" of a disjunction of Boolean-weighted counting functions, we prove that, for any constant c, over the domain Z_2^n, the class of disjunctions of no more than $\log\log n^c$ many Boolean-weighted counting functions with modulus q for a given integer $q \geq 2$ is polynomial time learnable with equivalence and membership queries.

2 Preliminaries

We assume that Z is the set of all integers. For any integer $n \geq 1$, let V_n be the set of variables $x_1, \ldots, x_n$. Let $Z_q = \{0, \ldots, q-1\}$ for any integer $q \geq 2$, $Z_q^n = \{0, \ldots, q-1\}^n$. Elements in Z_q^n are thought of here as $n \times 1$ vectors. We consider counting functions that consist of variables in V_n. Our example space will be Z^n and Z_q^n for $q \geq 2$. When $q = 2$, Z_q^n is the *n-bit* Boolean space. For any positive integer q, any $k \in Z_q$, and any integer vector $\vec{a} = (a_1, \ldots, a_n)^T \in Z^n$, an integer-weighted counting function $C_{q,k,\vec{a}}$ with modulus q is defined as

$$C_{q,k,\vec{a}}(x_1, \ldots, x_n) = \begin{cases} 0 & \text{if } \sum_{i=1}^n a_i x_i \equiv k \pmod q, \\ 1 & \text{otherwise.} \end{cases}$$

Here we say that $\vec{a}$ is the integer-weight vector (or weight for short) of $C_{q,k,\vec{a}}$. When $k = 0$, we say that $C_{q,0,\vec{a}}$ is an integer-weighted modulo function, and denote it by $M_{q,\vec{a}}$. When $\vec{a} \in Z_2^n$, we say that $C_{q,k,\vec{a}}$ (or $M_{q,\vec{a}}$) is a Boolean-weighted counting (or modulo) function.

For an integer-weighted counting function $C_{q,k,\vec{a}}$, let $vars(C_{q,k,\vec{a}})$ denote the set of all relevant variables x_i of $C_{q,k,\vec{a}}$, i.e., variables x_i such that $a_i \neq 0$. A disjunction F of integer-weighted counting functions $C_{q_1,k_1,\vec{a}_1}, \ldots, C_{q_t,k_t,\vec{a}_t}$ is $C_{q_1,k_1,\vec{a}_1} \vee \cdots \vee C_{q_t,k_t,\vec{a}_t}$. Let $vars(F)$ be the set of all relevant variables of F, i.e., the set $vars(C_{q_1,k_1,\vec{a}_1}) \cup \cdots \cup vars(C_{q_t,k_t,\vec{a}_t})$. If for any $i, j \in \{1, \ldots, t\}$, $i \neq j$ implies that $vars(C_{q_i,k_i,\vec{a}_i}) \cap vars(C_{q_j,k_j,\vec{a}_j}) = \phi$, then we say that F is read-once, i.e., each relevant variable of F occurs in exactly one counting function in F.

For $X \in \{Z_q, Z\}$, an example $\alpha \in X^n$ satisfies a counting function C if and only if $C(\alpha) = 1$. α is a positive example for a disjunction F of counting functions if it satisfies at least one counting function in F (we write $F(\alpha) = 1$) and a negative example otherwise (we write $F(\alpha) = 0$). For an example $\alpha \in Z_2^n$, let $\alpha[i]$ denote the i-th bit value of α, i.e., the value of the variable x_i in α. In general, for any literal y, $\alpha[y]$ denotes the value of y in α. For $i \in \{1, \ldots, n\}$, $flip(\alpha, i)$ stands for the example obtained from α by flipping exactly the i-th bit value in α. More generally, for a set $I \subseteq \{1, \ldots, n\}$, let $flip(\alpha, I)$ be the example obtained from α by flipping the i-th bit value in α for every $i \in I$. For convenience, we also extend $flip$ to act on literals or sets of literals in the following way, when $l \in \{x_i, \overline{x}_i\}$, let $flip(\alpha, l) = flip(\alpha, i)$, and similarly define $flip(\alpha, S)$ for a set S of literals.

Our learning model is the standard model for on-line learning with equivalence and membership queries (see, [A]). A learning process for a class C of Boolean-valued functions over the domain X^n with the variable set V_n is viewed as a dialogue between a learner A and the environment. The goal of the learner is to learn an unknown *target* function $f \in C$ that has been fixed by the environment. In order to gain information about f the learner proposes a hypothesis function h from a fixed hypothesis space H with $C \subseteq H$. Whenever $h \neq f$ for the proposed hypothesis h, the environment responds with a counterexample $\alpha \in X^n$ such that $h(\alpha) \neq f(\alpha)$. The learner may also ask membership queries for some examples $\alpha \in X^n$, to which the environment responds with "yes" if $f(\alpha) = 1$ or "no" if otherwise. The learner succeeds when he receives "yes" for an equivalence query from the environment, or he can conclude that the current hypothesis is logically equivalent to the target function f. We assume that the time complexity of asking a membership query for an example is the cost to write it down, and the time complexity of asking an equivalence query for hypothesis h is the cost to write h down. We

say that C is polynomial time learnable with equivalence and membership queries, if there is an algorithm for learning any target function $f \in C$, using polynomially in n and the size of f many equivalence and membership queries, while the time complexity of the algorithm is polynomial in n, the size of f, and the size of the largest example that occurred during the learning process.

3 Counting Functions via Linear Systems

In this section, we design algorithms for learning disjunctions of counting functions with a prime modulus. Our algorithms are based on the folklore algorithm for learning a vector space (see e.g. [HSW], [BCJ]). However, they are stronger than the folklore algorithm, because they make substantial improvement on the hypothesis representation: The hypotheses issued by our algorithms are disjunctions of no more $n+1$ counting functions, this is in contrast to the vector space hypotheses used by the folklore algorithm.

We assume that K is an arbitrary field; addition and multiplication of two elements in K, and inversion of a nonzero element in K, are all of polynomial time complexity. For any positive integer n, K^n is a vector space of dimension n over the field K. Every $\alpha \in K^n$ denotes an $n \times 1$ vector, and α^T is the $1 \times n$ transposition of α. Let $\vec{0}_{m,1}$ be an $m \times 1$ zero-vector, $\vec{X}_{n,1}$ be an $n \times 1$ vector of n variables $x_1, \ldots, x_n$, where x_i takes values from K. For any $m \times n$ matrix $A_{m,n}$ and any $m \times 1$ vector $\vec{b}_{m,1}$ over K, a linear system $L(A_{m,n}, \vec{b}_{m,1})$ of m linear equations over K is given as follows,

$$A_{m,n}\vec{X}_{n,1} = \vec{b}_{m,1}.$$

$\alpha \in K^n$ is a solution of the linear system $L(A_{m,n}, \vec{b}_{m,1})$, if

$$A_{m,n}\alpha = \vec{b}_{m,1}.$$

When $\vec{b}_{m,1} = \vec{0}_{m,1}$, we say that $L(A_{m,n}, \vec{b}_{m,1})$ is a homogeneous linear system, or homogeneous system for short. For convenience, we write $L(A_{m,n}) = L(A_{m,n}, \vec{0}_{m,1})$. The following two general theorems are established.

Theorem A. *The class of all homogeneous systems over the domain K^n for any given field K is polynomial time learnable with at most n equivalence queries. Moreover, the hypotheses issued by the learner are also homogeneous systems over K with no more than n linear equations.*

Theorem B. *The class of all linear systems over the domain K^n for any given field K is polynomial time learnable with at most $n+1$ equivalence queries. Moreover, the hypotheses issued by the learner are also linear systems over the field K with no more than n equations.*

For the rest of this section we assume that p is a given prime number, and $q \geq 2$ is a given integer. We know that Z_p is a field with modulo p addition and multiplication. Note that addition and multiplication of any two numbers in Z_p, and inversion of any non-zero number in Z_p, are of $poly(\log p)$ complexity, where the length of any number in Z_p is no more than $\log p$. Before we prove the above two general theorems, we first give several corollaries.

Corollary A.1. *Assume $q \geq 2$. Given $X \in \{Z_q, Z\}$, the class of all disjunctions of modulo functions $M_{p,\vec{a}}$ with integer-weights $\vec{a} \in Z^n$ over the domain X^n is polynomial time learnable with at most n equivalence queries, while the hypotheses issued by the learner are disjunctions of at most n modulo functions $M_{p,\vec{a}}$ with weights $\vec{a} \in Z_p^n$.*

Corollary A.2. *Given $X \in \{Z_q, Z\}$ with $q \geq 2$. Let $P = \{p_1, \ldots, p_k\}$ be a set of prime numbers. Then, the class of all disjunctions of modulo functions $M_{p,\vec{a}}$ with integer-weights $\vec{a} \in Z^n$ and $p \in P$ over the domain X^n is polynomial time learnable with at most kn equivalence queries, while the hypotheses issued by the learner are disjunctions of at most kn modulo functions $M_{p,\vec{a}}$ with weights $\vec{a} \in Z_p^n$ and $p \in P$.*

Corollary B.1. *Given $X \in \{Z_q, Z\}$ with $q \geq 2$. The class of all disjunctions of counting functions $C_{p,k,\vec{a}}$ with integer-weights $\vec{a} \in Z^n$ over the domain X^n is polynomial time learnable with at most $n+1$ equivalence queries, while the hypotheses issued by the learner are disjunctions of at most n counting functions $C_{p,k,\vec{a}}$ with weights $\vec{a} \in Z_p^n$.*

Corollary B.2. *Given $X \in \{Z_q, Z\}$ with $q \geq 2$. Let $P = \{p_1, \ldots, p_k\}$ be a set of prime numbers. Then, the class of all disjunctions of counting functions $C_{p,k,\vec{a}}$ with integer-weights $\vec{a} \in Z^n$ and $p \in P$ over the domain X^n is polynomial time learnable with at most $k(n+1)$ equivalence queries, while the hypotheses issued by the learner are disjunctions of at most kn counting functions $C_{p,k,\vec{a}}$ with weights $\vec{a} \in Z_p^n$ and $p \in P$.*

We now prove our theorems.

Proof of Theorem A. Assume that $L(A_{m,n})$ is the target system. Let $I_{l,l}$ be the $l \times l$ identity matrix over K. Let S_r be the set of all solutions received during the first r stages, the learning algorithm Learn-HS (where "HS" stands for "homogeneous system") is given as follows.

Algorithm Learn-HS:

Stage 1. Set the first hypothesis $H_1 = L(I_{n,n})$. Ask an equivalence query for H_1. If the learner receives "yes" then stop, otherwise he receives a non-zero solution $\vec{\alpha}_1 \in K^n$ to $L(A_{m,n})$. Let $S_1 = \{\vec{\alpha}_1\}$.

Stage $r \geq 2$. Let $S_{r-1} = \{\vec{\alpha}_1, \ldots, \vec{\alpha}_{r-1}\}$. Construct from vectors in S_{r-1} a matrix $B_{n-(r-1),n}$ such that the set of all solutions of the homogeneous system $L(B_{n-(r-1),n})$ is $span(S_{r-1}) = \{t_1\vec{\alpha}_1 + \cdots + t_{r-1}\vec{\alpha}_{r-1} | t_i \in K, 1 \leq i \leq r-1\}$. Set the r-th hypothesis $H_r = L(B_{n-(r-1),n})$. If $r = n+1$, the learner concludes that H_r is equivalent to $L(A_{m,n})$ so stop. When $r \leq n$, ask an equivalence query for H_r, if "yes" then stop, otherwise the learner re-

ceives a solution $\vec{\alpha}_r$ which is outside $span(S_{r-1})$. Set $S_r = S_{r-1} \cup \{\vec{\alpha}_r\}$.

End of Learn-HS.

Claim 3.1. *At any stage r with $1 < r \leq n+1$, the following holds: (1)vectors in S_{r-1} are linearly independent; (2)the matrix $B_{n-(r-1),n}$ exists; (3)$span(S_{r-1})$ is the set of all solutions of H_r; (4)every vector in $span(S_{r-1})$ is a solution of the target system.*

Proof of Claim 3.1. By induction on r. When $r = 2$, S_1 contains exactly one nonzero solution $\vec{\alpha}_1$ of the target system $L(A_{m,n})$, so it is trivial that vectors in S_1 are linearly independent, and every vector in $span(S_1)$ is a solution of $L(A_{m,n})$. Since $\vec{\alpha}_1$ is nonzero, we may assume without loss of generality that the first element in it is not 0. Let $\vec{\alpha}_1 = (a_{11}, a_{21}, \ldots, a_{n1})^T$. Since K is a field, $a_{11} \neq 0$ implies the inverse a_{11}^{-1} exists. Let $D_{n-1,1} = (a_{21}, \ldots, a_{n1})^T$ and define the matrix

$$B_{n-1,n} = \left(-D_{n-1,1}\left(a_{11}^{-1}\right), E_{n-1,n-1}\right).$$

Then, $B_{n-1,n}$ has rank $n-1$. By simple calculation, $B_{n-1,n}\vec{\alpha}_1 = \vec{0}_{n-1,1}$. Thus, $span(S_1)$ is exactly the set of all solutions of the system $L(B_{n-1,n})$. Hence, our claim holds for $r = 2$.

Assume our claim is true for any r with $1 < r \leq k < n+1$. At stage $k+1$, by the induction assumption, we know that vectors in S_{k-1} are linearly independent, vectors in $span(S_{k-1})$ are solutions of the target system, and $span(S_{k-1})$ is the set of all solutions of the hypothesis H_k. Thus, when the learner receives a counterexample $\vec{\alpha}_k$ for H_k, then $\vec{\alpha}_k$ is a solution of the target system outside $span(S_{k-1})$, this implies that $\vec{\alpha}_k$ is linearly independent from vectors in S_{k-1}. Hence, vectors in $S_k = S_{k-1} \cup \{\vec{\alpha}_k\}$ are linearly independent and vectors in $span(S_k)$ are solutions of the target system. Define the matrix $Q_{n,k} = (\vec{\alpha}_1, \ldots, \vec{\alpha}_k)$. Since K is a field, we may assume without loss of generality that the submatrix $G_{k,k}$ consisting of elements in the first k rows in $Q_{n,k}$ has an inverse $G_{k,k}^{-1}$. Let $N_{n-k,k}$ be the submatrix consisting of elements in the last $n-k$ rows in $Q_{n,k}$. Define the matrix

$$B_{n-k,n} = \left(-N_{n-k,k}G_{k,k}^{-1}, E_{n-k,n-k}\right).$$

Then, $B_{n-k,n}$ has rank $n-k$, and $B_{n-k,n}Q_{n,k} = \vec{0}_{n-k,k}$. Thus, $span(S_k)$ is the set of all solutions of the system $L(B_{n-k,n})$. Combining the above analysis, our claim holds. □

By the above claim, at any stage r with $2 \leq r \leq n$, either the learner learns the target system, or receives a solution of the target system which is linearly independent from the solutions in S_{r-1}. Since the target system has at most n linearly independent solutions, the learner learns it with at most n equivalence queries.

Let N be the size of of the longest element in any counterexamples received by the learner during the learning process. By the assumption that addition and multiplication of any two elements in K, and inversion of any element in K, are of polynomial time complexity, one can find at stage r the matrix $B_{n-(r-1),n}$ in time polynomial in n and N. So, the total time complexity of the algorithm Learn-HS is polynomial in n and N. □

Proof of Theorem B. Assume that $L(A_{m,n}, \vec{b}_{m,1})$ is the target system. Let $I_{l,l}$ be the $l \times l$ identity matrix over K. The learning algorithm Learn-IHS works in stages.

Algorithm Learn-IHS:

Stage 0. Choose a matrix $B_{n,n}$ and a vector $\vec{d}_{n,1}$ over K such that the rank of $B_{n,n}$ is different from that of the matrix $(B_{n,n}, \vec{d}_{n,1})$. Ask an equivalence query for the hypothesis $H_0 = L(B_{n,n}, \vec{d}_{n,1})$. Note that H_0 has no solutions. If the learner receives "yes" then stop, otherwise he receives a solution $\vec{\alpha}_0$ for the target system. Set $S_0 = \phi$.

Stage 1. Set the hypothesis $H_1 = L(I_{n,n}, \vec{\alpha}_0)$. Ask an equivalence query for H_1. If "yes" then stop, otherwise the learner receives a solution $\vec{\alpha}_1 \in K^n$ to $L(A_{m,n}, \vec{b}_{m,1})$ other than $\vec{a}_0$. Let $S_1 = \{\vec{\alpha}_1 - \vec{\alpha}_0\}$.

Stage $r \geq 2$. Let $S_{r-1} = \{\vec{\alpha}_1 - \vec{\alpha}_0, \ldots, \vec{\alpha}_{r-1} - \vec{\alpha}_0\}$. Construct from vectors in S_{r-1} a matrix $B_{n-(r-1),n}$ such that the set of all solutions of the homogeneous system $L(B_{n-(r-1),n})$ is $span(S_{r-1}) = \{t_1(\vec{\alpha}_1 - \vec{\alpha}_0) + \cdots + t_{r-1}(\vec{\alpha}_{r-1} - \vec{\alpha}_0) | t_i \in K, 1 \leq i \leq r-1\}$. Set the r-th hypothesis $H_r = L(B_{n-(r-1),n}, B_{n-(r-1),n}\vec{\alpha}_0)$. If $r = n+1$, the learner concludes that H_r is equivalent to $L(A_{m,n})$, so stop. If $r \leq n$, ask an equivalence query for H_r, if "yes" then stop, otherwise the learner receives a solution $\vec{\alpha}_r$. Set $S_r = S_{r-1} \cup \{\vec{\alpha}_r - \vec{\alpha}_0\}$.

End of Learn-IHS.

Claim 3.2. *At any stage r with $1 < r \leq n+1$, the following holds: (1)vectors in S_{r-1} are linearly independent; (2)the matrix $B_{n-(r-1),n}$ exists; (3)$span(S_{r-1})$ is the set of all solutions of the homogeneous system $L(B_{n-(r-1),n})$, and every vector $\vec{a} \in span(S_{r-1})$ is a solution of the homogeneous system $L(A_{m,n})$; (4)finally, $Ispan(S_{r-1}) = \{\vec{\alpha} + \vec{\alpha}_0 | \ \vec{\alpha} \in span(S_{r-1})\}$ is the set of all solutions of the hypothesis $H_r = L(B_{n-(r-1),n}, B_{n-(r-1),n}\vec{\alpha}_0)$, and any $\vec{\alpha} \in Ispan(S_r)$ is a solution of the target system $L(A_{m,n}, \vec{b}_{m,1})$.*

Proof of Claim 3.2. By induction on r. When $r = 2$, S_1 contains exactly one nonzero solution $\vec{\alpha}_1 - \vec{\alpha}_0$ of the homogeneous system $L(A_{m,n})$, since both $\vec{\alpha}_1$ and $\vec{\alpha}_0$ are solutions to the target system. So it is trivial that vectors in S_1 are linearly independent, i.e., (1) is true.

Since by (1) $\vec{\alpha}_1 - \vec{\alpha}_0$ is linearly independent, we may assume without loss of generality that the first element in it is not 0. Let $\vec{\alpha}_1 - \vec{\alpha}_0 = (a_{11}, a_{21}, \ldots, a_{n1})^T$. Since K is a field, $a_{11} \neq 0$ implies the inverse a_{11}^{-1} exists. Let $D_{n-1,1} = (a_{21}, \ldots, a_{n1})^T$. We can in fact write the

matrix $B_{n-1,n}$ as follows

$$B_{n-1,n} = \left(-D_{n-1,1}\left(a_{11}^{-1}\right), I_{n-1,n-1}\right).$$

This implies (2).

Note that $B_{n-1,n}$ has rank $n-1$. By simple calculation, $B_{n-1,n}(\vec{\alpha}_1 - \vec{\alpha}_0) = \vec{0}_{n-1,1}$. Thus, $span(S_1)$ is exactly the set of all solutions of the system $L(B_{n-1,n})$. Since each vector in S_{r-1} is a solution to $L(A_{m,n})$, so are vectors in $span(S_{r-1})$. Thus, (3) is true.

Note that $\vec{\alpha}_0$ is a solution to $L(B_{n-1,n}, B_{n-1,n}\vec{\alpha}_0)$. By (3), $Ispan(S_1)$ is the set of all solutions of $H_2 = L(B_{n-1,n}, B_{n-1,n}\vec{\alpha}_0)$, and every vector in $Ispan(S_1)$ is a solution to the target system. Hence, (4) is true.

Assume our claim is true for any r with $1 < r \leq k < n+1$. At stage $k+1$, by the induction assumption, we know that vectors in S_{k-1} are linearly independent, $span(S_{k-1})$ is the set of all solutions of the hypothesis H_k, and vectors in $Ispan(S_{k-1})$ are solutions of the target system. Thus, when the learner receives a counterexample $\vec{\alpha}_k$ for H_k, then $\vec{\alpha}_k$ is a solution of the target system outside $Ispan(S_{k-1})$. This implies that $\vec{\alpha}_k - \vec{\alpha}_0$ is linearly independent from vectors in S_{k-1}. Hence, vectors in $S_k = S_{k-1} \cup \{\vec{\alpha}_k - \vec{\alpha}_0\}$ are linearly independent, i.e., (1) is true.

Let the matrix $Q_{n,k} = ((\vec{\alpha}_1 - \vec{\alpha}_0), \ldots, (\vec{\alpha}_k - \vec{\alpha}_0))$. Since K is a field, we may assume without loss of generality that the submatrix $G_{k,k}$ consisting of elements in the first k rows in $Q_{n,k}$ has an inverse $G_{k,k}^{-1}$. Let $N_{n-k,k}$ be the submatrix consisting of elements in the last $n-k$ rows in $Q_{n,k}$. The matrix $B_{n-k,n}$ exists and in fact we can write it as

$$B_{n-k,n} = \left(-N_{n-k,k}G_{k,k}^{-1}, \quad I_{n-k,n-k}\right).$$

Hence, (2) is true.

$B_{n-k,n}$ has rank $n-k$, and $B_{n-k,n}Q_{n,k} = \vec{0}_{n-k,k}$. Thus, by (1), $span(S_k)$ is the set of all solutions of the homogeneous system $L(B_{n-k,n})$, and each vector in $span(S_k)$ is a solution to $L(A_{m,n})$. This implies that (3) is true.

Note that $\vec{\alpha}_0$ is a solution to $L(B_{n-k,n}, B_{n-k,n}\vec{\alpha}_0)$. By (3), $Ispan(S_1)$ is the set of all solutions of $H_{k+1} = L(B_{n-k,n}, B_{n-k,n}\vec{\alpha}_0)$, and every vector in $Ispan(S_k)$ is a solution to the target system. Hence, (4) is true. □

By Claim 3.2, at any stage r with $1 < r \leq n$, either the learner learns the target system, or receives a solution $\vec{\alpha}_r$ of the target system such that $\vec{\alpha}_r - \vec{\alpha}_0$ is linearly independent from the solutions in S_r. Since the homogeneous system $L(A_{m,n})$ of the target system has at most n linearly independent solutions, so the learner learns $L(A_{m,n})$ (and hence $L(A_{m,n}, \vec{b}_{m,1})$) with at most $n+1$ equivalence queries. Since addition and multiplication of any two elements in K, and inversion of any nonzero element in K, are of polynomial time complexity, at any stage r, one can find the matrix $B_{n-(r-1),n}$ in time polynomially in n and the size of the longest element in any vectors received during the first r stages. Thus, the time complexity of the algorithm Learn-IHS is polynomial in n and the size of the longest element in vectors received during the learning process. □

Proof of Corollary A.1. Assume $F = M_{p,\vec{a}_1} \vee \cdots \vee M_{p,\vec{a}_t}$ is the target function. For the integer-weight $\vec{a}_i = (a_{i1}, \ldots, a_{in})^T$ of $M_{p,\vec{a}_i}$, let $\vec{b}_i = (a_{i1} \bmod p, \ldots, a_{in} \bmod p)^T$. Then, $\vec{b}_i \in Z_p^n$, and F is equivalent to the function

$$F^* = M_{p,\vec{b}_1} \vee \cdots \vee M_{p,\vec{b}_t}.$$

Hence, in order to learn F, one only needs to learn F^*. Define a matrix

$$A_{t,n} = \begin{pmatrix} (\vec{b}_1)^T \\ \vdots \\ (\vec{b}_t)^T \end{pmatrix}.$$

Then, F^* (and hence F) is equivalent to the homogeneous system over the domain Z_q^n

$$A_{t,n}\vec{X}_{n,1} = \vec{0}_{t,1}$$

in the sense that, for any vector $\vec{x} \in Z_q^n$, $F^*(\vec{x}) = 0$ if and only if $\vec{x}$ is a solution of the above system. Note also that for a vector $\vec{d} \in Z_p^n$, a linear equation $(\vec{d})^T \cdot \vec{X}_{n,1} \equiv 0 \pmod p$ is equivalent to the modulo function $M_{p,\vec{d}}$. Therefore, our corollary follows from Theorem A and Lemma 3.3 and 3.4. □

Lemma 3.3. *Assume $q \leq p$. Let $L(B_{s,n})$ be a given homogeneous system over the domain Z_p^n with modulo p addition and multiplication. Assume that $S = \{\vec{\alpha}_1, \ldots, \vec{\alpha}_r\}$ is a set of linearly independent vectors in Z_p^n such that $span(S) = \{k_1\vec{\alpha}_1 + \ldots + k_r\vec{\alpha}_r | k_i \in Z_p, 1 \leq i \leq r\}$ is the set of all solutions of $L(B_{s,n})$ over the domain Z_p^n. Then, the set of all solutions of $L(B_{s,n})$ over the domain Z_q^n is $Rspan(S) = span(S) \cap Z_q^n$.*

Proof. It is obvious that any vector in $Rspan(S)$ is a solution of $L(B_{s,n})$ over Z_q^n. Suppose that $\vec{b}$ is a solution of $L(B_{s,n})$ over Z_q^n, then it is also a solution of $L(B_{s,n})$ over Z_p^n, since $Z_q^n \subseteq Z_p^n$. Hence, $\vec{b} \in Rspan(S)$. □

Lemma 3.4. *Given $X \in \{Z_q, Z\}$ with $q > p$. Let $L(B_{s,n})$ be a homogeneous system over the domain X^n with modulo p addition and multiplication. Assume that $S = \{\vec{\alpha}_1, \ldots, \vec{\alpha}_r\}$ is a set of linearly independent vectors in Z_p^n such that $span(S) = \{k_1\vec{\alpha}_1 + \ldots + k_r\vec{\alpha}_r | k_i \in Z_p, 1 \leq i \leq r\}$ is the set of all solutions of $L(B_{s,n})$ over the domain Z_p^n. Then, the set of all solutions of $L(B_{s,n})$ over the domain X^n is $Espan(S) = \{\vec{\alpha} + (q_1p, \ldots, q_np)^T | \vec{\alpha} \in span(S); q_j \in X, 1 \leq j \leq n\} \cap X^n$.*

Proof. It is obvious that any vector in $Espan(S)$ is a solution of $L(B_{s,n})$ over X^n. Suppose that $\vec{b} = (b_1, \ldots, b_n)^T \in X^n$ is a solution of $L(B_{s,n})$. Let $b_j = d_j + q_jp$, where $d_j \in Z_p, q_j \in X$. Let $\vec{d} = (d_1, \ldots, d_n)^T$, then $\vec{d} \in Z_p^n$, and $\vec{d}$ is a solution of $L(B_{s,n})$ because $\vec{b}$ is. Since $span(S)$ is the set of all solutions of $L(B_{s,n})$ over Z_p^n, there exist $k_i \in Z_p$ for $1 \leq i \leq r$ such that $\vec{d} =$

$k_1\vec{\alpha}_1 + \cdots + k_r\vec{\alpha}_r$. Thus, $\vec{b} = k_1\vec{\alpha}_1 + \cdots + k_r\vec{\alpha}_r + (q_1p, \ldots, q_np)^T \in Espan(S)$. □

Proof of Corollary A.2. Assume that

$$F = M_{p_1,\vec{a}_{11}} \vee \cdots \vee M_{p_1,\vec{a}_{1t_1}} \vee \cdots \vee M_{p_k,\vec{a}_{k1}} \vee \cdots \vee M_{p_k,\vec{a}_{kt_k}}$$

is a target function. For the integer-weight $\vec{a}_{ij} = (b_{ij1}, \ldots, b_{ijn})^T$ of $M_{p_i,\vec{a}_{ij}}$, let

$$\vec{a}^*_{ij} = (b_{ij1} \bmod p_i, \ldots, b_{ijn} \bmod p_i)^T.$$

Then, $\vec{a}^*_{ij} \in Z^n_{p_i}$, and F is equivalent to the function

$$F^* = M_{p_1,\vec{a}^*_{11}} \vee \cdots \vee M_{p_1,\vec{a}^*_{1t_1}} \vee \cdots \vee M_{p_k,\vec{a}^*_{k1}} \vee \cdots \vee M_{p_k,\vec{a}^*_{kt_k}}.$$

Hence, in order to learn F, one only needs to learn F^*. Define the matrices,

$$A^i_{t_i,n} = \begin{pmatrix} (\vec{a}^*_{i1})^T \\ \vdots \\ (\vec{a}^*_{it_i})^T \end{pmatrix}, \quad i = 1, \ldots, k.$$

Then, F^* (and hence F) is equivalent to the "conjunction" of the homogeneous systems

$$A^i_{t_i,n}\vec{X}_{n,1} = \vec{0}_{t_i,1}, \quad i = 1, \ldots, k,$$

over the domain X^n with modulo p_i addition and multiplication in the sense that, for any vector $\vec{u} \in X^n$, $F^*(\vec{u}) = 0$ if and only if $\vec{u}$ is a solution for each of the above systems. Note also that for a vector $\vec{u} \in Z^n$, a linear equation $(\vec{u})^T \cdot \vec{X}_{n,1} \equiv 0 \pmod{p_i}$ is equivalent to the modulo function $M_{p_i,\vec{u}}$ with the integer-weight $\vec{u}$.

One then learns F^* (hence F) through learning $L(A^i_{t_i,n})$, for $i = 1, \ldots, k$, simultaneously. At each stage, let H_i be the hypothesis for $L(A^i_{t_i,n})$, $i = 1, \ldots, k$. In other word, H_i is a hypothesis for

$$M_{p_i,\vec{a}^*_{i1}} \vee \cdots \vee M_{p_i,\vec{a}^*_{it_i}}.$$

One sets $H = H_1 \vee \cdots \vee H_k$ to be the hypothesis for F^*. According to Corollary A.1 one can learn each of the systems $L(A^i_{t_i,n})$ with at most n equivalence queries, and the hypotheses issued by the learner are homogeneous systems with weights from Z_{p_i}. When one receives a counterexample for the hypothesis H, one can derive from this counterexample a new linearly independent vector (i.e., solution) for at least one of the systems $L(A^i_{t_i,n})$. Thus, with at most kn equivalence queries one can learn F^*. Since by Corollary A.1 the time complexity for learning each of the systems $L(A^i_{t_i,n})$ is polynomial in n and the largest size of elements in vectors received by the learner during the learning process, so the time complexity for learning F^* is $kP(n, N)$, where P is a polynomial and N is the size of the largest element in any vectors received by the learner. □

Proof of Corollary B.1. Assume $F = C_{p,k_1,\vec{a}_1} \vee \cdots \vee C_{p,k_t,\vec{a}_t}$ is the target function. Our proof is similar to that of Corollary A.1. But, instead of modulo functions $M_{p,\vec{a}_i}$, we consider counting functions $C_{p,k_i,\vec{a}_i}$, $i = 1, \ldots, t$. In the same manner as we did for Corollary A.1, we obtain a matrix $A_{t,n}$. Let $R_{t,1} = (k_1, \ldots, k_t)^T$. Then, F is equivalent to the linear system over the domain Z^n_q

$$A_{t,n}\vec{X}_{n,1} = \vec{R}_{t,1}.$$

Therefore, our corollary follows from Theorem B and Lemma 3.5 and 3.6 □

Lemma 3.5. *Assume $q \le p$. Let $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$ be a given linear system over the domain Z^n_p. Assume that $S = \{\vec{\alpha}_1, \ldots, \vec{\alpha}_r\}$ is a set of linearly independent vectors in Z^n_p such that $span(S) = \{k_1\vec{\alpha}_1 + \ldots + k_r\vec{\alpha}_r | k_i \in Z_p, 1 \le i \le r\}$ is the set of all solutions of $L(B_{s,n})$ over the domain Z^n_p. Then, the set of all solutions of $L(B_{s,n})$ over the domain Z^n_q is $Rspan(S) = span(S) \cap Z^n_q$, and the set of all solutions of $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$ over the domain Z^n_q is $IRspan(S) = Ispan(S) \cap Z^n_q$, where $Ispan(S) = \{\vec{\alpha} + \vec{b}_{s,1} | \vec{\alpha} \in span(S)\}$.*

Proof. It is obvious that any vector in $Rspan(S)$ is a solution of $L(B_{s,n})$ over Z^n_q, and any vector in $IRspan(S)$ is a solution of $L(B_{s,n}, \vec{b}_{s,1})$ over Z^n_q. Suppose that $\vec{f}$ is a solution of $L(B_{s,n})$ over Z^n_q, then it is also a solution of $L(B_{s,n})$ over Z^n_p, since $Z^n_q \subseteq Z^n_p$. Hence, $\vec{f} \in Rspan(S)$. When $\vec{g}$ is a solution of $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$ over Z^n_q, then $\vec{g} - \vec{b}_{s,1}$ is a solution of $L(A_{m,n})$ over the domain Z^n_p. Thus, $\vec{g} - \vec{b}_{s,1} \in span(S)$, which implies $\vec{g} \in Ispan(S)$. Hence, $\vec{g} \in IRspan(S)$. □

Lemma 3.6. *Given $X \in \{Z_q, Z\}$ with $q > p$. Let $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$ be a linear system system over the domain X^n with modulo p addition and multiplication. Assume $S = \{\vec{\alpha}_1, \ldots, \vec{\alpha}_r\}$ is a set of linearly independent vectors in Z^n_p such that $span(S) = \{k_1\vec{\alpha}_1 + \ldots + k_r\vec{\alpha}_r | k_i \in Z_p, 1 \le i \le r\}$ is the set of all solutions of $L(B_{s,n})$ over the domain Z^n_p. Then, the set of all solutions of $L(B_{s,n})$ over the domain X^n is $Espan(S) = \{\vec{\alpha} + (q_1p, \ldots, q_np)^T | \vec{\alpha} \in span(S);\ q_j \in X, 1 \le j \le n\} \cap X^n$, and the set of all solutions of $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$ is $IEspan(S) = \{\vec{f} + \vec{b}^*_{s,1} | \vec{f} \in Espan(S)\}$. Here, $b_{s,1} = (b_1, \ldots, b_s)^T$, $b^*_{s,1} = (b_1 \bmod p, \ldots, b_s \bmod p)^T$.*

Proof. It is obvious that any vector in $Espan(S)$ is a solution of $L(B_{s,n})$ over X^n, and any vector in $IEspan(S)$ is a solution of $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$ over X^n. Suppose that $\vec{f} = (f_1, \ldots, f_n)^T \in X^n$ is a solution of $L(B_{s,n})$. Let $f_j = d_j + q_jp$, where $d_j \in Z_p, q_j \in X$. Let $\vec{d} = (d_1, \ldots, d_n)^T$, then $\vec{d} \in Z^n_p$, and $\vec{d}$ is a solution of $L(B_{s,n})$ because $\vec{f}$ is. Since $span(S)$ is the set of all solutions of $L(B_{s,n})$ over Z^n_p, there exist $k_i \in Z_p$ for $1 \le i \le r$ such that $\vec{d} = k_1\vec{\alpha}_1 + \cdots + k_r\vec{\alpha}_r$. Thus, $\vec{f} = k_1\vec{\alpha}_1 + \cdots + k_r\vec{\alpha}_r + (q_1p, \ldots, q_np)^T \in Espan(S)$. Similarly, when $\vec{g} \in X^n$ is a solution of $L(B_{s,n}, B_{s,n}\vec{b}_{s,1})$, $\vec{g} \in IEspan(S)$. □

Proof of Corollary B.2. Assume that

$$F = C_{p_1,k_{11},\vec{a}_{11}} \vee \cdots \vee C_{p_1,k_{1t_1},\vec{a}_{1t_1}} \vee \cdots \vee C_{p_s,k_{s1},\vec{a}_{s1}} \vee \cdots \vee C_{p_s,k_{st_s},\vec{a}_{st_s}}$$

is a target function. Instead of modulo functions $M_{p_i,\vec{a}_i,j}$ in the proof of Corollary A.2, we now consider counting functions $C_{p_i,k_{ij},\vec{a}_{ij}}$. Thus, we obtain matrices $A^i_{t_i,n}$ in the same manner. Define

$$\vec{R}_{t_i,1} = (k_{11}, \ldots, k_{it_i})^T, i = 1, \ldots, s.$$

Then, F is equivalent to the "conjunction" of the linear systems,

$$A^i_{t_i,n}\vec{X}_{n,1} = \vec{R}_{t_i,1}, \quad i = 1, \ldots, s,$$

over the domain X^n with modulo p_i addition and multiplication in the sense that, for any vector $\vec{u} \in X^n$, $F^*(\vec{u}) = 0$ if and only if $\vec{u}$ is a solution to each of the above systems. Hence, our corollary follows from Corollaries B.1, with a similar analysis to the proof of Corollary A.2. □

4 Read-Once Disjunctions of Counting Functions

As argued in [BGHM], it is reasonable to believe that an equivalence query is more expensive than a membership query. A practically ideal learning algorithm will use as few equivalence queries as possible. We will design a learning algorithm for the class of read-once disjunctions of Boolean-weighted counting functions over the domain Z_2^n that requires only *one* (it is not hard to see that this is also the lower bound) equivalence query. Previous work ([BHH]) shows that this class can be learned using equivalence and membership queries, but the bound on the number of equivalence queries is n^3. In the following, we assume that $q \geq 2$ is a given integer, $F = C_{q,k_1,\vec{a}_1} \vee \cdots \vee C_{q,k_t,\vec{a}_t}$ is a disjunction of counting functions with Boolean-weights $\vec{a}_i \in Z_2^n$, $i = 1, \ldots, t$. We also assume that α is a negative counterexample for F.

Lemma 4.1. *For any variable $x \in vars(C_{q,k_i,\vec{a}_i})$, $F(flip(\alpha, x)) = C_{q,k_i,\vec{a}_i}(flip(\alpha, x)) = 1$, $i = 1, \ldots, t$.*

Proof. Since α is a negative example for F, $C_{q,k_i,\vec{a}_i}(\alpha) = 0$. This implies that

$$S = \sum_{x \in vars(C_{q,k_i,\vec{a}_i})} \alpha[x] \equiv k_i \pmod{q}.$$

Hence, for any $x \in vars(C_{q,k_i,\vec{a}_i})$, after flipping x in α, the original sum S modulo q then becomes either $k_i + 1$ or $k_i - 1$, so $F(flip(\alpha, x)) = C_{q,k_i,\vec{a}_i}(flip(\alpha, x)) = 1$. □

Lemma 4.2. $vars(F) = \{x \in V_n | F(flip(\alpha, x)) = 1\}$.

Proof. On the one hand, by Lemma 4.1, $vars(F) = \bigcup\{vars(C_{q,k_i,\vec{a}_i}) | i = 1, \ldots, t\} \subseteq \{x \in V_n | F(flip(\alpha, x)) = 1\}$. On the other hand, for any variable $y \in \{x \in V_n | F(flip(\alpha, x)) = 1\}$, we have $C_{q,k_j,\vec{a}_j}(flip(\alpha, y)) = 1$ for some $j \in \{1, \ldots, t\}$. Note again that $C_{q,k_j,\vec{a}_j}(\alpha) = 0$, since α is a negative example. Thus, $y \in vars(C_{q,k_i,\vec{a}_i})$ and, $vars(F) = \{x \in V_n | F(flip(\alpha, x)) = 1\}$. □

Lemma 4.3. *For any two distinct variables $u, v \in vars(C_{q,k_i,\vec{a}_i})$, for any $w \notin vars(C_{q,k_i,\vec{a}_i})$, $1 \leq i \leq t$, we have (1)$F(flip(\alpha, \{u, w\})) = 1$ and, (2)$F(flip(\alpha, \{u, v\})) = 0$ if $\alpha[u] \neq \alpha[v]$.*

Proof. It follows from $F(\alpha) = 0$ that $C_{q,k_i,\vec{a}_i}(\alpha) = 0$, i.e.,

$$S = \sum_{x \in vars(C_{q,k_i,\vec{a}_i})} \alpha[x] \equiv k_i \pmod{q}.$$

For $u \in vars(C_{q,k_i,\vec{a}_i})$ and $w \notin vars(C_{q,k_i,\vec{a}_i})$, after flipping u and w in α, the above sum S is changed to $k_i - 1 \bmod q$ or $k_i + 1 \bmod q$, thus $F(flip(\alpha, \{u, w\})) = C_{q,k_i,\vec{a}_i}(flip(\alpha, \{u, w\})) = 1$. For two distinct variables $u, v \in vars(C_{q,k_i,\vec{a}_i})$, if $\alpha[u] \neq \alpha[v]$, after flipping u and v in α, the above sum S is still $k_i \bmod q$, thus $F(flip(\alpha, \{u, w\})) = C_{q,k_i,\vec{a}_i}(flip(\alpha, \{u, v\})) = 0$. □

Lemma 4.4. *Assume that F is read-once. Then, for any set S of exactly p variables such that they all have the same value in α, $F(flip(\alpha, S)) = 0$ if and only if $S \subseteq vars(C_{q,k_i,\vec{a}_i})$ for some $C_{q,k_i,\vec{a}_i}$ in F.*

Proof. The sufficient condition is trivial, since F is read-once. Assume $F(flip(\alpha, S)) = 0$ and suppose by contradiction that $S \not\subseteq vars(C_{q,k_i,\vec{a}_i})$ for any $C_{q,k_i,\vec{a}_i}$ in F, this implies that there are $C_{q,k_i,\vec{a}_i}$ and $C_{q,k_j,\vec{a}_j}$ with $i \neq j$ such that $S \cap vars(C_{q,k_i,\vec{a}_i}) \neq \phi$, and $S \cap vars(C_{q,k_j,\vec{a}_j}) \neq \phi$. Thus, $F(flip(\alpha, S)) = C_{q,k_i,\vec{a}_i}(flip(\alpha, S)) = C_{q,k_i,\vec{a}_i}(flip(\alpha, S)) = 1$, a contradiction. So, there must be some $C_{q,k_i,\vec{a}_i}$ in F such that $S \subseteq vars(C_{q,k_i,\vec{a}_i})$. □

Lemma 4.5. *Assume $vars(C_{q,k_i,\vec{a}_i}) = \{u_1, \ldots, u_m\}$ and $m < q$. Then, (1)$C_{q,k_i,\vec{a}_i}$ is equivalent to $[C_{q,0,\vec{a}_i}(u_1) \vee \cdots \vee C_{q,0,\vec{a}_i}(u_m)]$ if $[\alpha[u_i] = \cdots = \alpha[u_m] = 0]$; (2)$C_{q,k_i,\vec{a}_i}$ is equivalent to $[C_{q,1,\vec{a}_i}(u_1) \vee \cdots \vee C_{q,1,\vec{a}_i}(u_m)]$ if $[\alpha[u_i] = \cdots = \alpha[u_m] = 1]$.*

Proof. Note that $C_{q,k_i,\vec{a}_i}(\alpha) = 0$. When $\alpha[u_i] = \cdots = \alpha[u_m] = 0$, $\alpha[u_1] + \cdots + \alpha[u_m] = 0 \equiv k_i \pmod{q}$. When $\alpha[u_i] = \cdots = \alpha[u_m] = 1$, $\alpha[u_1] + \cdots + \alpha[u_m] = m \equiv k_i \pmod{q}$. In the first case, we have $k_i = 0$. Since $m < q$, $C_{q,0,\vec{a}_i}(u_i, \ldots, u_m)$ is equivalent to $C_{q,0,\vec{a}_i}(u_1) \vee \cdots \vee C_{q,0,\vec{a}_i}(u_m)$. In the latter case, we have $k_i = m < q$, thus $C_{q,m,\vec{a}_i}(u_i, \ldots, u_m)$ is equivalent to $C_{q,1,\vec{a}_i}(u_1) \vee \cdots \vee C_{q,1,\vec{a}_i}(u_m)$. □

Theorem 4.1. *The class of all read-once disjunctions of Boolean-weighted counting functions with modulus q over the domain Z_2^n is polynomial time learnable using only one equivalence query and $O(n^q)$ membership queries.*

Proof. Assume $F = C_{q,k_1,\vec{a}_1} \vee \cdots \vee C_{q,k_t,\vec{a}_t}$ is the target function. We construct the learning algorithm Learn-RODC (where "RODC" stands for "read-once disjunctions of counting functions") that runs in stages.

Algorithm Learn-RODC:

Stage 0. Ask an equivalence query for the "TRUE" function. If "yes" then stop; otherwise the learner receives a negative counterexample α.

Stage 1. For each $x \in V_n$, ask a membership query

for $flip(\alpha, x)$. Let $vars(F)$ be the set of all those x such that the learner receives "yes" for $flip(\alpha, x)$.

Stage 2. Fix any $u \in vars(F)$. For any $v \in vars(F) - \{u\}$ such that $\alpha[u] \neq \alpha[v]$, ask a membership query for $flip(\alpha, \{u, v\})$. Let G_u be the set of all those v such that the learner receives "no" for $flip(\alpha, \{u, v\})$. Let P_u be the set of all those x such that $G_x = G_u \neq \phi$, and $\alpha[x] = \alpha[u]$. Set $PG = \{(P_u, G_u) | u \in vars(F), G_u \neq \phi\}$.

Stage 3. Let $Rvars(F)$ be the set of all variables in $vars(F)$ but not in any set in PG. Fix any $u \in Rvars(F)$. For any subset S of $Rvars(F) - \{u\}$ with exactly $q - 1$ variables such that all those variables and u have the same value in α, ask a membership query for $flip(\alpha, \{u\} \cup S)$. Let S_u be the union of all those subsets S and $\{u\}$ such that the learner receives "no" for $flip(\alpha, \{u\} \cup S)$. Set $RS = \{S_u | u \in Rvars(F), S_u \neq \phi\}$.

Stage 4. Let $Evars(F)$ be the set of all variables in $vars(F)$ but not in any sets in PG or RS. For any set $A \subseteq V_n$, let $\vec{a}(A)$ be the characteristic vector of A, and $k(A) = \sum_{x \in A} \alpha[x] \bmod q$. The learner concludes that the target function F is equivalent to $H = \bigvee\{C_{q,k(P \cup G),a(P \cup G)} | (P, G) \in PG\} \vee$
$\bigvee\{C_{q,k(S),\vec{a}(S)} | S \in RS\} \vee$
$\bigvee\{C_{q,\alpha[x],\vec{a}(\{x\})} | x \in Evars(F)\}$.

End of Learn-RODC.

We now analyze the algorithm Learn-RODC. We may assume without loss of generality that $F \not\equiv$ "$TRUE$". Thus, at stage 0, the learner receives a negative counterexample α for F. It follows from Lemma 4.2 that one finds $vars(F)$ at stage 1 with n membership queries. At stage 2, by Lemma 4.3, one finds all those $vars(C_{q,k_i,\vec{a}_i})$ such that there are two variables in $vars(C_{q,k_i,\vec{a}_i})$ with different values in α. Thus, $\bigvee\{C_{q,k(P \cup G),a(P \cup G)} | (P, G) \in PG\}$ is the disjunction of all those counting functions in F such that each of them has two relevant variables with different values in α. The number of membership queries required at this stage is at most $2n^2$. At stage 3, by Lemma 4.4, one finds all those $vars(C_{q,k_i,\vec{a}_i})$ such that $vars(C_{q,k_i,\vec{a}_i})$ consists of at least p variables that have the same value in α. Thus, $\bigvee\{C_{q,k(S),\vec{a}(S)} | S \in RS\}$ is the disjunction of all those counting functions in F such that each of them has at least p relevant variables with the same value in α. The number of membership queries required at this stage is at most n^q. By Lemma 4.5, $\bigvee\{C_{q,\alpha[x],\vec{a}(\{x\})} | x \in Evars(F)\}$ is equivalent to the disjunction of all those counting functions $C_{q,k_i,\vec{a}_i}$ in F such that $vars(C_{q,k_i,\vec{a}_i})$ consists of less than p relevant variables that have the same value in α. No membership queries are required at this stage. With the above analysis, F is equivalent to H. Learn-RODC needs only one equivalence query and $n + 2n^2 + n^q$ membership queries. The time complexity is $O(n^2 + 2n^3 + n^{q+1}) = O(n^{q+1})$. □

5 Disjunctions of a Non-Constant Number of Counting Functions

A typical strategy for learning k-term DNF formulas with equivalence and membership queries is that at each stage the learner tries to learn only one term in the target formulas while turning all the other terms off. The difficulty involved in this strategy is how the learner can turn all terms off except one. When $k = O(\log n)$, it was overcome by Blum and Rudich's derandomization technique [BR]. However, unlike a monomial which turns on if and only if all its literals turn on, a counting function depends on the modulo p value of the sum of its variables. Thus, it is not hard to see that Blum and Rudich's technique are not suitable for learning a disjunction of a non-constant number of counting functions. Nevertheless, based on analyzing the "modulo-structure" of counting functions, we prove that for any constant c, any disjunction with no more than $\log \log n^c$ many Boolean-weighted counting functions over the domain Z_2^n is polynomial time learnable.

Assume that $q \geq 2$ is a given integer number, $F = C_{q,k_1,\vec{a}_1} \vee \cdots \vee C_{q,k_t,\vec{a}_t}$ is a disjunction of counting functions over the domain Z_2^n with Boolean-weights $\vec{a}_i \in Z_2^n$. Assume also that α is a negative counterexample for F. For any $S \subseteq vars(F)$, define $C_S = \{C_{q,k_i,\vec{a}_i} | S \subseteq vars(C_{q,k_i,\vec{a}_i}), 1 \leq i \leq t\}$. We say that $S \neq \phi$ is a "modulo-block" of F if, $S = \bigcap_{C_{q,k_i,\vec{a}_i} \in C_S} vars(C_{q,k_i,\vec{a}_i})$, and for any $C_{q,k_j,\vec{a}_j} \notin C_S$, $S \cap vars(C_{q,k_j,\vec{a}_j}) = \phi$. Let MB_F ("MB" stands for "modulo-blocks") denote the set of all modulo-blocks of F. Note that For any two modulo-blocks $B, D \in MB_F$, either $B = D$, or $B \cap D = \phi$.

Lemma 5.1. *For any modulo-block $B \in MB_F$, for any two distinct variables $x, y \in B$ and, for any variable $u \in vars(F) - B$, we have (1)$F(flip(\alpha, \{x, u\})) = 0$ and, (2)$F(flip(\alpha, \{x, y\})) = 0$ if $\alpha[x] \neq \alpha[y]$.*

Proof. By the definition, $x, y \in B$ implies $x, y \in vars(C_{q,k_i,\vec{a}_i})$ for any $C_{q,k_i,\vec{a}_i} \in C_B$ and $x, y \notin vars(C_{q,k_j,\vec{a}_j})$ for any $C_{q,k_j,\vec{a}_j} \notin C_B$. $C_{q,k_i,\vec{a}_i}(\alpha) = 0$ means that

$$\sum_{v \in vars(C_{q,k_i,\vec{a}_i})} \alpha[v] \equiv k_i \pmod{q}.$$

If $\alpha[x] \neq \alpha[y]$, the above sum will not change after flipping both x and y in α. So, $F(flip(\alpha, \{x, y\})) = C_{q,k_i,\vec{a}_i}(flip(\alpha, \{x, y\})) = 0$. On the other hand, it is easy to see that $C_{q,k_i,\vec{a}_i}(flip(\alpha, \{x\})) = 1$ for any $C_{q,k_i,\vec{a}_i} \in C_B$. Since $u \notin B$, there is a $C_{q,k_j,\vec{a}_j} \in C_B$ such that $u \notin vars(C_{q,k_j,\vec{a}_j})$. Hence, $F(flip(\alpha, \{x, w\})) = C_{q,k_j,\vec{a}_j}(flip(\alpha, \{x, u\})) = C_{q,k_j,\vec{a}_j}(flip(\alpha, \{x\})) = 1$. □

Lemma 5.2. *For any $S \subseteq vars(F)$ with exactly p variables such that they all have the same value in α, $F(flip(\alpha, S)) = 0$ if and only if $S \subseteq B$ for some modulo-block $B \in MB_F$.*

Proof. The sufficient condition is trivial by the definition of modulo-blocks. Assume $F(flip(\alpha, S)) = 0$ and

suppose by contradiction that S is not a subset of any modulo-blocks of F. This implies that there are two distinct modulo-blocks B_1 and B_2 in MB_F such that $S \cap B_1 \neq \phi$ and $S \cap B_2 \neq \phi$. Hence, by the definition of modulo-blocks, there is one counting function in C_{B_1} and another in C_{B_2} such that each of them has at least one but less than p variables of S. So, after flipping all variables in S in α, those two counting functions (thus F) will have value 1, a contradiction to the earlier assumption. □

Lemma 5.3. *For any counting function $C_{q,k_i,\vec{a}_i}$ in F, there are modulo-blocks $B_1, \ldots, B_m \in MB_F$ such that $\vec{a}_i$ is the characteristic vector of $B = B_1 \cup \cdots \cup B_m$, $k_i = \sum_{x \in B} \alpha[x] \bmod q$.*

Proof. We first show that there are modulo-blocks $B_1, \ldots, B_m \in MB_F$ such that $vars(C_{q,k_i,\vec{a}_i}) = B_1 \cup \cdots \cup B_m$. Fix a variable $x_1 \in vars(C_{q,k_i,\vec{a}_i})$. Let

$$Q_1 = \bigcap \{vars(C_{q,k_j,\vec{a}_j}) | x_1 \in vars(C_{q,k_j,\vec{a}_j})\}.$$

Then, $x_1 \in Q_1$. Define $B_1 = \{y \in Q_1 | \forall C_{q,k_j,\vec{a}_j} \notin Q_1, y \notin vars(C_{q,k_j,\vec{a}_j})\}$. It is easy to see that, $x \in B_1$, and B_1 is a modulo-block of F. Note that $B_1 \subseteq vars(C_{q,k_i,\vec{a}_i})$. If $B_1 = vars(C_{q,k_i,\vec{a}_i})$, then we are done. Otherwise, fix a variable $x_2 \in vars(C_{q,k_i,\vec{a}_i}) - B_1$. We define Q_2 and B_2 in the same manner, thus we obtain a new modulo-block B_2 with $x_2 \in B_2 \subseteq vars(C_{q,k_i,\vec{a}_i})$. If $B_1 \cup B_2 = vars(C_{q,k_i,\vec{a}_i})$, then we are done. Otherwise, repeat the above process to obtain a new modulo-block. Note that $vars(C_{q,k_i,\vec{a}_i})$ contains at most n variables. We eliminate at least one variable from $vars(C_{q,k_i,\vec{a}_i})$ when we obtain a new modulo-block. Thus, we have m modulo-blocks $B_1, \ldots, B_m$, $m \leq n$, such that $vars(C_{q,k_i,\vec{a}_i}) = B_1 \cup \cdots \cup B_m$, $m \leq n$. It then follows that $\vec{a}_i$ is the characteristic vector of $B = B_1 \cup \cdots \cup B_m$. $C_{q,k_i,\vec{a}_i}(\alpha) = 0$ implies that $k_i = \sum_{x \in B_1 \cup \ldots \cup B_m} \alpha[x] \bmod p$. □

Lemma 5.4. $\|MB_F\| \leq 2^t$. *In other words, F has at most 2^t modulo-blocks.*

Proof. According to Lemma 5.3, given a negative counterexample for F, each $C_{q,k_i,\vec{a}_i}$ in F is determined by the modulo-blocks that consist of $vars(C_{q,k_i,\vec{a}_i})$. Thus, we can represent F with a matrix M, M has t rows and m columns. The i-th row of M stands for the the function $C_{q,k_i,\vec{a}_i}$. Each column contains a modulo-block, and no two columns have the same modulo-block. Let $e_{i,j}$ denote the entry of M at the i-th row and the j-th column. Assume that the j-th column contains the modulo-block B_j. Then $e_{i,j} = B_j$ if $B_j \subseteq vars(C_{q,k_i,\vec{a}_i})$, otherwise let $e_{i,j}$ = *"blank"*. We now estimate how large t can be. For column a and column b, $a \neq b$, by the definition of modulo-blocks, there exists at least one i such that $e_{i,a}$ differs from $e_{i,b}$, i.e., either $e_{i,a} = B_a$ but $e_{i,b}$ = *"blank"*, or $e_{i,a}$ = *"blank"* but $e_{i,b} = B_b$. This implies that $m \leq 2^t$, since there are at most 2^t many possible ways to place a modulo-block in a column. Thus, $\|MB_F\| \leq 2^t$. □

Theorem 5.1. *There is an algorithm for learning the class of disjunctions of no more than* $\log\log n^c$ *many Boolean-weighted counting functions with modulus q over the domain Z_2^n, using $O(n^q + n^{c(q+1)})$ many queries. The time complexity of the algorithm is bounded by $O(n^{c+1} + n^{2c(q+1)+1})$. So for constant c, the algorithm is polynomial.*

Proof. Assume that $F = C_{q,k_1,\vec{a}_1} \vee \cdots \vee C_{q,k_t,\vec{a}_t}$ is the target function. The learning algorithm runs in stages.

At stage 0, the learner issues the initial hypothesis H_1 = *"TRUE"* to ask an equivalence query. If he receives "yes" then stop. Otherwise, he receives a negative example α for F. One query is used at this stage, the time complexity is constant.

At stage 1, for any $x \in V_n$, the learner asks a membership query for $flip(\alpha, x)$. By Lemma 4.2, the learner finds $vars(F)$, i.e., the set of all those variables such that flipping any one of them in α will cause F to output 1. The number of queries used at this stage is n, the time complexity is $O(n^2)$.

At stage 2, using Lemma 5.1, the learner finds all those modulo-blocks such that each of them has two distinct variables with different values in α: For any $u \in vars(F)$ and $v \in vars(F) - \{u\}$ such that u and v have different values in α, ask a membership query for $flip(\alpha, \{u, v\})$. Let $A(u)$ be the set of all those v such that the learner receives "no". Let $E(u)$ be the set of all those w such that $A(w) = A(u) \neq \phi$ and $\alpha[w] = \alpha[u]$. Set $B_u = A_u \cup E_u$, then B_u is a modulo-block. At this stage at most n^2 membership queries are required and the time complexity is $O(n^3)$.

At stage 3, using Lemma 5.2, the learner finds all those modulo-blocks such that each of them has at least q variables and all of the variables in it have the same value in α: For any $u \in vars(F)$, for any set $S \subseteq vars(F) - \{u\}$ with exactly $q - 1$ variables such that u and variables in S have the same value in α, ask a membership query for $flip(\alpha, \{u\} \cup S)$. Let $S(u)$ be the union of all those subsets S and $\{u\}$ such that the learner receives "no" for $flip(\alpha, \{u\} \cup S)$, then $S(u)$ is a modulo-block if it is not empty. The number of queries used at this stage is at most n^q, and the time complexity is $O(n^{q+1})$.

At stage 4, the learner finds all *possible* modulo-blocks such that each of them has at most $q - 1$ variables and all variables in it have the same value in α: Let FB be the set of all modulo-blocks found at the above stages 2 and 3 and RB be the set of all variables in $vars(F)$ but not in any modulo-blocks in FB. Then, each modulo-block $B \in MB_F - FB$ has less than q variables and all variables in it have the same value in α. It is trivial that B is a subset of RB. By Lemma 5.4, $\|RB\| \leq q2^t$. Actually, one finds RB as a by-product of stage 2 and stage 3, i.e., whenever one finds a modulo-block at those two stages one eliminates all variables in it from $vars(F)$. The remaining variables in $vars(F)$ is RB. Thus, the number of queries required at this stage is 0, the time complexity is $O(n^3 + n^{q+1})$.

At stage 5, the learner constructs all possible counting

functions using modulo-blocks in FB and subsets in RB: For any modulo-blocks $B_1, \ldots, B_m \in FB$, for any subset R of RB, set $W = B_1 \cup \cdots \cup B_m \cup R$. Define a counting function $H(B_1, \ldots, B_m, R)$ as $C_{q,l,\vec{a}}$, where $\vec{a}$ is the characteristic vector of W, and $l = \sum_{x \in W} \alpha[x] \bmod q$. Finally, the learner sets the hypothesis

$$H_2 = \bigvee_{B_1,\ldots,B_m \in MB, R \subseteq MR} H(B_1, \ldots, B_m, R).$$

With Lemma 5.3, every counting function in F is contained in H_2. The number of queries required at this stage is 0, the time complexity is $O(n2^{2^t}2^{q2^t})$.

At stage 6, the learner asks equivalence queries for the hypothesis H_2. If the answer is "yes" then stop. Otherwise one receives a negative counterexample β, since H_2 contains all counting functions in F. Thus, one eliminates every counting functions in H_2 that outputs 1 for β. One still uses H_2 to denote the disjunction of the remaining counting functions in H_2. Repeat the above process until one receives "yes". The number of queries used at this stage is at most $2^{2^t}2^{q2^t}$, since H_2 originally contains at most $2^{2^t}2^{q2^t}$ counting functions. For each equivalence query one needs to write down the hypothesis, so the time complexity of this stage is at most $O(n2^{2^{t+1}}2^{q2^{t+1}})$.

Combining the above analysis, the learner needs $O(n^q + 2^{2^t}2^{q2^t})$ many queries to learn F, and the time complexity is bounded by $O(n^{q+1} + n2^{2^{t+1}}2^{q2^{t+1}})$. When $t \leq \log\log n^c$, the number of queries is bounded by $O(n^q + n^{c(q+1)})$, and the time complexity is bounded by $O(n^{q+1} + n^{2c(q+1)+1})$. □

6 Concluding Remarks

Negations. We don't know whether disjunctions of integer-weighted counting functions with a prime modulus are still polynomial time learnable when some functions are negated. In particular, we don't know whether disjunctions of negations of integer-weighted counting functions are polynomial time learnable. Currently, we prove that disjunctions of integer-weighted counting functions are still polynomial time learnable if they contain constant number of negated counting functions.

Composite Moduli. We don't know whether disjunctions of integer-weighted counting functions with a composite modulus are polynomial time learnable. In particular, we don't know whether disjunctions of Boolean-weighted counting functions over the Boolean domain are polynomial time learnable. Very recently, Jeffrey Jackson [J] observed from Fourier analysis that the class of disjunctions of $O(\log n)$ parities is polynomial time learnable. It might be possible to extend his result to the class of disjunctions of $O(\log n)$ counting functions with a composite modulus.

Acknowledgments. We are very grateful to Robert Schapire for valuable discussions on this topic, and especially for providing us with his algorithm for learning disjunctions of parities with equivalence queries. The algorithms Learn-HS and Learn-IHS are motivated by Schapire's parity-learning algorithm and by the algorithm V developed by Helmbold, Sloan and Warmuth [HSW].

References

[A] D. Angluin, "Queries and concept learning", *Machine Learning*, 2, 1988, pages 319-342.

[AHK] D. Angluin, L. Hellerstein, M. Karpinsky, "Learning learning read-once formulas with queries", *J. ACM*, 1, 1993, pages 185-210.

[BR] A. Blum, S. Rudich, "Fast learning of k-term DNF formulas with queries", *Proc of the 24th Annual ACM Symposium on Theory of Computing*, May 1992, pages 382-389.

[BCJ] A. Blum, P. Chalasani, J. Jackson, "On learning embedded symmetric concepts", *Proc of the Sixth Annual ACM Conference on Computational Learning Theory*, pages 337-346, Morgan Kaufmann Publishers, Inc.,San Mateo, CA, 1993.

[BS] A. Blum, M. Singh, "Learning functions of k terms", *Proc of the Third Annual Workshop on Computational Learning Theory*, pages 144-153, Morgan Kaufmann Publishers, Inc.,San Mateo, CA, 1990.

[BGHM] N. Bshouty, S. Goldman, T. Hancock, S. Matar, "Asking queries to minimize errors", *Proc of the 6th Annual ACM Conference on Computational Learning Theory*, pages 41-50, 1993.

[BHH] N. Bshouty, T. Hancock, L. Hellerstein, "Learning boolean read-once formulas with arbitrary symmetric and constant fan-in gates", *Proc of the 5th Annual Workshop on Computational Learning Theory*, pages 1-15, Morgan Kaufmann Publishers, Inc.,San Mateo, CA, 1992.

[FS] P. Fisher, H. Simon, "On learning ring-sum-expansions", *SIAM J. Comput.*, 1992, pages 181-192.

[HH] T. Hancock, L. Hellerstein, "Learning read-once formulas over fields and extended bases", *Proc of the 3th Annual Workshop on Computational Learning Theory*, pages 326-336, 1991.

[HSW] D. Helmbold, R. Sloan, M. Warmuth, "Learning integer lattices", *SIAM J. Comput.*, 1992, pages 240-266.

[J] J. Jackson, Personal communications.

[S] R. Schapire, Personal communications.