The Exact Multiplicative Complexity of the Hamming Weight Function

Joan Boyar (joan@imada.sdu.dk) ⋆[1] René Peralta (peralta@cs.yale.edu) ⋆⋆[2]

[1] Department of Mathematics and Computer Science
University of Southern Denmark, Odense, Denmark
[2] Department of Computer Science
Yale University

Abstract. We consider the problem of computing the Hamming weight of an n-bit vector using a circuit with gates for addition and multiplication modulo 2 (alternatively, XOR and conjunction gates) only. The number of multiplications necessary and sufficient to build such a circuit is called the "multiplicative complexity" of the Hamming weight function, and is denoted by $c_\wedge(H^n)$. We prove $c_\wedge(H^n) = n - H^{\mathbb{N}}(n)$ where $H^{\mathbb{N}}(n)$ is the Hamming weight of the binary representation of n.

1 Introduction

The *multiplicative complexity* $c_\wedge(f)$ of a Boolean function f is the number of conjunctions necessary and sufficient to implement a circuit which computes f over the basis $(\wedge, \oplus, 1)$ (alternatively, the number of multiplications necessary and sufficient to calculate a function over GF_2 via a straight-line program). Let $\overrightarrow{H}(\mathbf{x})$ denote the binary representation of the Hamming weight of a bit string $\mathbf{x} \in GF_2^n$. $\overrightarrow{H}(\mathbf{x})$ has fixed length $\lceil \log_2(n+1) \rceil$ and may contain leading zeros. The function $\overrightarrow{H}()$ will be denoted by H^n when the parameter n needs to be explicitly stated. We will denote by $H^{\mathbb{N}}(n)$ the Hamming weight of the binary representation of the integer n. In this paper we show

$$c_\wedge(H^n) = n - H^{\mathbb{N}}(n).$$

Our motivation to study multiplicative complexity comes from cryptography. In [2] a construction is given for a non-interactive cryptographic proof of knowledge of a secret X. The secret is defined as the input to a public Boolean circuit C, where the output C(X) is also publicly known. Such constructions are called "discreet proofs", and have a number of applications, e.g. electronic voting, on-line auctions, fair exchange of secret keys, etc. The length of a discreet proof is linear in the number

⋆ Partially supported by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT), and by the Danish Natural Science Research Council (SNF).

⋆⋆ Partially supported by NSF grant CCR-0081823.

of conjunctions in C and is not affected by the number of negations or disjunctions. Computation of the Hamming weight is of particular interest because it is often an intermediate step for constructing efficient circuits for use in discreet proofs.

Our result is also of interest from a purely complexity-theoretic point of view. For measures of circuit complexity in general, the exact complexity of explicitly defined Boolean functions is known only for a small number of simple functions. For most functions, only weak lower bounds are known [5]. To our knowledge, the only other exact solutions for the multiplicative complexity of non-trivial functions appear in [1], [6], and [4]. The first two papers relate the multiplicative complexity of an arbitrary quadratic form to the rank of an associated matrix over GF_2. The third paper contains the result $c_{\wedge}(\Sigma_3^n) = \lceil \frac{n}{2} \rceil$, where Σ_k^n is the k^{th} elementary symmetric function on n variables. [1]

2 Preliminaries

There is a bijection between Boolean functions on n variables and square-free polynomials over GF_2^n. It is known that a Boolean function $f(\mathbf{x})$ whose polynomial has degree d has multiplicative complexity at least $d-1$ (for a proof see [3]). This we call the *degree lower bound.*

It will prove useful to define the Hamming weight of the empty string λ to be 0, i.e. $\overrightarrow{H}(\lambda) = H^{\mathbb{N}}(0) = 0$. We now make some simple observations.

- If $0 \leq i < 2^k$ then $H^{\mathbb{N}}(2^k + i) = 1 + H^{\mathbb{N}}(i)$. (1)
- If $0 \leq k$ then $H^{\mathbb{N}}(2^k - 1) = k$. (2)
- If $0 \leq a, b, k$ and $n = 2^k - 1 = a + b$ then $H^{\mathbb{N}}(n) = H^{\mathbb{N}}(a) + H^{\mathbb{N}}(b)$. (3)
- $\overrightarrow{H}(\mathbf{x})$ is the integer sum of the bits of $\mathbf{x}$.
- For all $n \geq m > 0$, there exists a circuit which adds an n-bit number to an m-bit number – plus an optional carry-in bit c – using n conjunctions. This is a standard addition circuit using a chain of full adders. A full adder computes the two-bit sum $w_1 w_0$ of three bits b_1, b_2, b_3. Only one conjunction is needed because $w_0 = (b_1 + b_2 + b_3) \bmod 2$ and $w_1 = ((b_1 + b_2)(b_2 + b_3) + b_2) \bmod 2$. We will refer to this circuit as the *standard addition circuit* (with carry-in c).

Denote by $c_{\wedge}(ADD(n, m))$ the multiplicative complexity of adding an n-bit number to an m-bit number. An immediate application of the degree lower bound is that $c_{\wedge}(ADD(n, m)) \geq Max(n, m)$. This is because $c_{\wedge}(ADD(n, m)) \geq c_{\wedge}(ADD(n, 1))$, and the most significant bit of this sum is the product of all $n+1$ input variables. We have already observed that $c_{\wedge}(ADD(n, m)) \leq Max(n, m)$. Thus we have shown

[1] $\Sigma_k^n(\mathbf{x})$ (or simply Σ_k^n) is defined by

$$\Sigma_k^n(x_1, x_2, \ldots, x_n) = \sum_{S \subseteq \{1, \ldots, n\}, |S| = k} \prod_{i \in S} x_i \qquad (1 \leq k \leq n).$$

Lemma 1. *The multiplicative complexity of adding two integer inputs, of lengths n and m in radix-2 representation, is $Max(n, m)$.*

3 A circuit for the Hamming weight

We construct a circuit for H^n that uses $n - H^{\mathbb{N}}(n)$ conjunctions. Our construction is essentially a recursive version of a construction that appeared in [3]. First we show a circuit for the case $n = 2^k - 1$.

Lemma 2. *Let $n = 2^k - 1$ for $k \geq 0$. Then $c_\wedge(H^n) \leq n - H^{\mathbb{N}}(n) = 2^k - (k+1)$.*

Proof. The proof is by induction on k. The cases $k = 0, 1$ are easily verifiable. For $k > 1$, a string $\mathbf{x}$ of length $2^k - 1$ can be split into two strings $\mathbf{u}, \mathbf{v}$, of length $2^{k-1} - 1$ each, plus one string c of length 1. We recursively compute $\overrightarrow{H}(\mathbf{u})$ and $\overrightarrow{H}(\mathbf{v})$. Then we use the standard addition circuit with carry-in c to compute $c + \overrightarrow{H}(\mathbf{u}) + \overrightarrow{H}(\mathbf{v})$. The result is $\overrightarrow{H}(\mathbf{x})$. By induction, and the fact that $\overrightarrow{H}(\mathbf{u}), \overrightarrow{H}(\mathbf{v})$ are of length $k - 1$, the number of multiplications used is $2(2^{k-1} - k) + k - 1 = 2^k - (k+1)$. □

We now consider the general case

Theorem 1. *$c_\wedge(H^n) \leq n - H^{\mathbb{N}}(n)$, for all $n \geq 1$.*

Proof. We have already shown this for the cases $n = 0, 1, 3, 7, 15, 31, \ldots$. We prove the remaining cases by induction on n. Let $\mathbf{x}$ be a string of length $2^k + i$ with $k > 0$ and $0 \leq i < 2^k - 1$. Assume the theorem holds for all values $0 \leq n' < 2^k + i$. As in Lemma 2 we split $\mathbf{x}$ into three strings $\mathbf{u}, \mathbf{v}, c$ of lengths $2^k - 1$, i, and 1 respectively (note that $\mathbf{v}$ may be the empty string). We recursively compute $\overrightarrow{H}(\mathbf{u})$ and $\overrightarrow{H}(\mathbf{v})$. Then we compute the sum $c + \overrightarrow{H}(\mathbf{u}) + \overrightarrow{H}(\mathbf{v})$. The result is $\overrightarrow{H}(\mathbf{x})$. By induction, using Lemma 2 and the fact that $\overrightarrow{H}(\mathbf{u}), \overrightarrow{H}(\mathbf{v})$ are of maximum length k, the number of multiplications used is

$$2^k - (k+1) + (i - H^{\mathbb{N}}(i)) + k = 2^k + i - (1 + H^{\mathbb{N}}(i)) = (2^k + i) - H^{\mathbb{N}}(2^k + i).$$

The last equality is due to observation (1). □

Lemma 11 of [3] shows that when $m = 2^i \leq n$, the value of $\Sigma^n_m(\mathbf{x})$ is the $i+1$st bit of $\overrightarrow{H}(\mathbf{x})$.[2] Suppose $\mathbf{x}$ is a bit string of length 2^k. The $k+1$st bit of $\overrightarrow{H}(\mathbf{x})$ is $\Sigma^{2^k}_{2^k}(\mathbf{x})$, which is a polynomial of degree 2^k. Thus, by the degree lower bound, it is not possible to compute the Hamming weight of a string of length 2^k bits using less than $2^k - 1$ multiplications. Since Theorem 1 gives a matching upper bound, we have

Corollary 1. $c_\wedge(H^{2^k}) = 2^k - H^{\mathbb{N}}(2^k) = 2^k - 1$ *for all $k \geq 0$.*

[2] See also [7].

4 The exact complexity of the Hamming weight

We proceed to show that the bound in Theorem 1 is tight, and hence the construction is optimal for all n.[3] The proof uses the known value of $c_\wedge(H^{2^k})$ to compute a lower bound on $c_\wedge(H^{2^k-i})$. For notational brevity, we will denote $c_\wedge(H^n)$ by h_n.

Theorem 2. *$c_\wedge(H^n) = n - H^{\mathbb{N}}(n)$, for all $n \geq 1$.*

Proof. By Corollary 1, we only need to consider the cases where n is strictly between consecutive powers of 2, i.e. $2^{k-1} < n < 2^k$. Our proof is by induction on k with base $k = 1$. Let $k > 1$ and assume the theorem holds for all $n' \leq 2^{k-1}$. Let $n = 2^k - i$ for some integer $1 \leq i < 2^{k-1}$. Then $n + (i-1) = 2^k - 1$ implies, by observation (3), that $k - H^{\mathbb{N}}(i-1) = H^{\mathbb{N}}(n)$. We design a circuit for the Hamming weight of a string $\mathbf{x}$ of length $2^k = n + (i-1) + 1$ as follows. We split $\mathbf{x}$ into three strings $\mathbf{u}, \mathbf{v}, c$ of lengths $n, i-1$, and 1, respectively. We use optimal circuits to compute $\overrightarrow{H}(\mathbf{u})$ and $\overrightarrow{H}(\mathbf{v})$. Note that the longest of these two strings is $\overrightarrow{H}(\mathbf{u})$, which has length k. Then we use the standard addition circuit with carry-in c to compute $c + \overrightarrow{H}(\mathbf{u}) + \overrightarrow{H}(\mathbf{v})$. The result is $\overrightarrow{H}(\mathbf{x})$. By the inductive hypothesis, the circuit for $\overrightarrow{H}(\mathbf{v})$ contains $h_{i-1} = (i-1) - H^{\mathbb{N}}(i-1)$ multiplications. Thus the circuit for $\overrightarrow{H}(\mathbf{x})$ contains $h_n + (i-1) - H^{\mathbb{N}}(i-1) + k$ multiplications. By Corollary 1, this quantity must be at least $2^k - 1$, i.e.

$$h_n + (i-1) - H^{\mathbb{N}}(i-1) + k \geq 2^k - 1.$$

Substituting $H^{\mathbb{N}}(n)$ for $k - H^{\mathbb{N}}(i-1)$, n for $2^k - i$, and rearranging terms, we obtain $h_n \geq n - H^{\mathbb{N}}(n)$. This lower bound matches the upper bound of Theorem 1. □

References

1. A. A. Aleksanyan. On realization of quadratic Boolean functions by systems of linear equations. *Cybernetics*, 25(1):9–17, 1989.
2. J. Boyar, I. Damgård, and R. Peralta. Short non-interactive cryptographic proofs. *Journal of Cryptology*, 13:449–472, 2000.
3. J. Boyar, R. Peralta, and D. Pochuev. On the multiplicative complexity of Boolean functions over the basis $(\wedge, \oplus, 1)$. *Theoretical Computer Science*, 235:43–57, 2000.
4. J. Boyar, R. Peralta, and D. Pochuev. Concrete multiplicative complexity of symmetric functions. Technical Report YALEU/DCS/TR1219, Computer Science Department, Yale University, 2001.
5. K. Lenz and I. Wegener. The conjunctive complexity of quadratic Boolean forms. *Theoretical Computer Science*, 81:257–268, 1991.

[3] This is quite surprising. In fact, we mistakenly stated in [3] that this bound was not tight.

6. R. Mirwald and C. Schnorr. The multiplicative complexity of quadratic Boolean forms. *Theoretical Computer Science*, 102(2):307–328, 1992.
7. R. Rueppel and J. Massey. The knapsack as a nonlinear function. In *Abstracts of papers, IEEE Int. Symp. on Information Theory*, page 46, 1985.