

# A Lower Bound for Primality\*

ERIC ALLENDER<sup>†</sup>

Dept. of Computer Science  
Rutgers University  
Piscataway, NJ 08854-8019, USA  
allender@cs.rutgers.edu

MICHAEL SAKS<sup>‡</sup>

Mathematics Dept.  
Rutgers University  
Piscataway, NJ 08854-8019, USA  
saks@math.rutgers.edu

IGOR SHPARLINSKI<sup>§</sup>

Dept. of Computing  
Macquarie University  
NSW 2109, Australia  
igor@comp.mq.edu.au

## Abstract

Recent work by Bernasconi, Damm and Shparlinski showed that the set of square-free numbers is not in  $AC^0$ , and raised as an open question whether similar (or stronger) lower bounds could be proved for the set of prime numbers. In this note, we show that the Boolean majority function is  $AC^0$ -Turing reducible to the set of prime numbers (represented in binary). From known lower bounds on MAJ (due to Razborov and Smolensky) we conclude that primality cannot be tested in  $AC^0[p]$  for any prime  $p$ . Similar results are obtained for the set of square-free numbers, and for the problem of computing the greatest common divisor of two numbers.

---

\*A preliminary version of this work appeared in Proc. 14th Annual IEEE Conference on Computational Complexity, 1999, pp. 10–14.

<sup>†</sup>Supported in part by NSF grant CCR-9734918.

<sup>‡</sup>Supported in part by NSF grant CCR-9700239.

<sup>§</sup>Supported in part by ARC grant A69700294.

# 1 Introduction

What is the computational complexity of the set of prime numbers? There is a large body of work presenting important upper bounds on the complexity of the set of primes (including [AH87, APR83, Mil76, R80, SS77]), but – as was pointed out recently in [BDS98a, BDS98b, BDS99, BS99, Shp98], very little has been published regarding *lower bounds* on the complexity of this set. Meidânis [Med91] proved an  $\Omega(\sqrt{\log n})$  time lower bound that holds in both the algebraic decision tree model with the floor function and the RAM model. Mantzivis [Man92] showed that the set of primes is not in the class of  $\Delta_0^b$ -definable sets, which is a uniform sub-class of  $AC_0$  that arises in the study of bounded arithmetic. In the context of space-bounded computation, it was shown in [HS68] that at least logarithmic space is required, in order to determine if a number is prime, under some plausible number-theoretic conjectures. Later on, it was shown in [HB76] that logarithmic space is both necessary and sufficient if the number is presented in *unary*. (It seems to still be unknown if the set of primes (encoded in binary) can be recognized in  $o(\log n)$  space.) However, these bounds do not address circuit complexity at all; note for instance that the *unary* encoding of prime numbers has trivial circuit complexity. Prior to the current work, it was not known whether primality testing was in  $AC^0$ , i.e., could be accomplished by constant-depth, polynomial-size circuits of AND, OR, and NOT gates. Independently, it was shown in [LV99] that primality testing is not in  $AC^0$ , under some unproved number-theoretic assumptions. Our method is different and, accordingly, our results are unconditional.

Recall that if  $F$  is a family of Boolean functions  $AC^0[F]$  is the class of functions computable by constant-depth polynomial-size circuits using AND, OR, and NOT gates and gates that compute functions from  $F$ . Recall that the Boolean majority function MAJ is defined to be 1 if and only if at least half of the input bits are 1 and the circuit class  $TC^0$  is defined to be  $AC^0[\text{MAJ}]$ . Also for a natural number  $k$ , the Boolean function  $\text{MOD}_k$  is defined to be 1 precisely if  $k$  divides the sum of the input bits and the circuit class  $AC^0[k]$  is defined to be  $AC^0[\text{MOD}_k]$ .

Our results concern three number-theoretic functions. In each case, the input is an integer or sequence of integers given in binary.

- PRIMES is the set of prime integers.
- SQUARE-FREE is the set of integers  $x$  that are not divisible by any perfect square greater than 1.

- GCD is the set of triples  $(x, y, i)$  of integers such that the  $i^{\text{th}}$  bit of the greatest common divisor of  $x$  and  $y$  is 1.

In this note, we prove:

**Theorem 1**  $TC^0$  is contained in each of the classes  $AC^0[\text{PRIMES}]$ ,  $AC^0[\text{GCD}]$  and  $AC^0[\text{SQUARE-FREE}]$ .

It is well-known that for any  $k$ ,  $\text{MOD}_k \in TC^0$  (i.e., for  $n$  bit input,  $\text{MOD}_k = \bigvee_{i=0}^{\lfloor n/k \rfloor} (\Theta_{ik} \wedge \neg \Theta_{ik+1})$  where  $\Theta_j$  is the function that is 1 if at least  $j$  input bits are 1.) Thus our results imply that  $\text{MOD}_k$  belongs to each of the classes  $AC^0[\text{PRIMES}]$ ,  $AC^0[\text{GCD}]$  and  $AC^0[\text{SQUARE-FREE}]$ . A fundamental result of Smolensky [Smo87] (building on earlier work of Razborov [Raz87]) says that if  $p$  and  $q$  are distinct primes, and  $\{C_n : n \in \mathbb{N}\}$  is a sequence of bounded depth circuits using AND, OR, NOT and  $\text{MOD}_p$  gates such that  $C_n$  computes  $\text{MOD}_q(x)$  on inputs of size  $n$ , then the size of  $C_n$  is exponential in  $n^\delta$  for some constant  $\delta > 0$ .

From this we conclude:

**Corollary 2** For any prime  $p$ , any circuit family of bounded depth of AND, OR, NOT and  $\text{MOD}_p$  gates that computes PRIMES, GCD, or SQUARE-FREE for  $n$  bit inputs has size exponential in  $n^\delta$  for some  $\delta > 0$ . In particular these three languages are not in  $AC^0[p]$ .

For the case of SQUARE-FREE and GCD our results strengthen and simplify some of the results of [BDS98a, BDS98b, BDS99, BS99] which, in particular, imply that these two languages are not in  $AC^0$ .

## 2 Preliminaries

### 2.1 Languages, functions and circuits

For a string  $x \in \{0, 1\}^*$ ,  $|x|$  denotes the length of  $x$ . Since we are dealing with number-theoretic functions and our strings often represent integers, it is convenient to index the bits of our strings by  $x = x_{n-1} \dots x_1 x_0$  so that the integer represented by  $x$  is  $\sum_i x_i 2^i$ .

Throughout the paper  $\log z$  and  $\ln z$  denote the binary and natural logarithms of  $z > 0$ , respectively.

A Boolean function family  $f$  is a sequence  $\{f_n : n \in \mathbb{N}\}$  of Boolean functions where for some function  $l(n) = l_f(n)$  that is bounded by a polynomial

in  $n$ ,  $f_n$  maps  $\{0, 1\}^{l(n)}$  to  $\{0, 1\}$ . (The typical case is  $l(n) = n$ , but it is often convenient to allow other functions). We follow the common abuse of notation that  $f$  can denote both the family of functions or a single function  $f_n$  in the class. We make the usual association between languages over  $\{0, 1\}^*$  and function families.

A *circuit family*  $C$  is a set  $\{C_n : n \in \mathbb{N}\}$  where each  $C_n$  is an acyclic circuit with  $l(n) = l_C(n)$  Boolean inputs  $x_{l(n)-1}, \dots, x_0$  (where  $l(n)$  is bounded by some polynomial function of  $n$ ), and some number,  $r_C(n)$ , of outputs.  $\{C_n\}$  has *size*  $s(n)$  if each circuit  $C_n$  has at most  $s(n)$  gates; it has *depth*  $d(n)$  if the length of the longest path from input to output in  $C_n$  is at most  $d(n)$ .

A function (family)  $f$  is said to be in (non-uniform)  $\text{AC}^0$  if there is a circuit family  $\{C_n\}$  of size  $n^{O(1)}$  and depth  $O(1)$  consisting of unbounded fan-in AND and OR and NOT gates such that for each  $n$ ,  $C_n$  computes  $f_n$  on inputs of length  $l_f(n)$ .

We note the following well-known facts:

**Proposition 3** *The following functions can be computed by bounded depth circuits of size polynomial in  $n$ :*

1. *Any function with at most  $\log n$  input bits and  $n^{O(1)}$  output bits.*
2. *The sum or difference of two integers of at most  $n^{O(1)}$  bits.*
3. *The product of a  $\log n$  bit integer and an  $n$  bit integer.*

The first fact is obtained by writing each output bit as a DNF formula, which can be represented by a depth 2 circuit of size at most  $k2^k$  where  $k$  is the number of input bits. The second fact can be found in many elementary books on circuit design or circuit complexity (for example, see [Str], pp. 127–128). For the third, Chandra, Stockmeyer and Vishkin (see [CSV84], p. 431) showed that multiplication of two  $n$  bit numbers is  $\text{AC}^0$ -reducible to the function that counts the number of ones in an  $n$ -bit string. Their argument shows that for  $k \leq n$ , the multiplication of an  $n$  bit string by a  $k$  bit number is  $\text{AC}^0$ -reducible to the function that counts the number of ones in a  $k$ -bit string. Since this latter function is in  $\text{AC}^0$  for  $k = \log n$ , the former is as well.

## 2.2 Reducibility

A language  $A_1$  is  $\leq_m^{\text{AC}^0}$  *reducible* to a language  $A_2$ , written  $A_1 \leq_m^{\text{AC}^0} A_2$ , if there is a function  $f$  in  $\text{AC}^0$  such that, for all  $x$ ,  $x \in A_1$  if and only if

$f(x) \in A_2$ .

$A_1$  is  $\leq_T^{\text{AC}^0}$  reducible to  $A_2$ , written  $A_1 \leq_T^{\text{AC}^0} A_2$  if  $A_1$  is recognized by a (nonuniform) family of circuits of polynomial size and constant depth, consisting of NOT gates, unbounded fan-in AND and OR gates, and oracle gates for  $A_2$ . (An oracle gate for  $A_2$  takes  $m$  inputs  $x_1, \dots, x_m$  and outputs 1 if  $x_1 \dots x_m$  is in  $A_2$ , and outputs 0 otherwise.) We write  $\text{AC}^0[L]$  for the class of languages  $A$  satisfying  $A \leq_T^{\text{AC}^0} L$ . It is well-known that  $\leq_T^{\text{AC}^0}$  is a transitive relation on languages.

Note that  $\leq_m^{\text{AC}^0}$  reducibility is a special case of  $\leq_T^{\text{AC}^0}$  reducibility.

The motivation for studying  $\leq_m^{\text{AC}^0}$  reducibility stems from the fact that most computational problems that arise in practice turn out to be complete for some well-known complexity class under  $\leq_m^{\text{AC}^0}$  reducibility. Number-theoretic problems are a notable exception; almost no number-theoretic problems are known to be complete for natural complexity classes.

### 2.3 The density of primes in arithmetic progressions

We need a technical fact about the density of primes within a given arithmetic progression. For integers  $1 \leq l < k \leq N$ , let  $\pi(N, k, l)$  denote the number of primes  $q$  such that  $q \leq N$  and  $q \equiv l \pmod{k}$ .

**Lemma 4** *There exists an integer  $P_0$  such that for any prime  $p \geq P_0$  and any integers  $N$  and  $l$  with  $N \geq 2^p$  and  $1 \leq l \leq p - 1$ ,*

$$\pi(N, p, l) \geq \frac{N}{2(\log N)^2}$$

This is obtained from Theorems 1 and 2 of [Pa35]. Theorem 1 gives a lower bound on  $\pi(N, k, l)$  whose error term depends on a technical function  $\sigma(k)$ . Theorem 2 gives an upper bound on  $\sigma(k)$ . Combining these two theorems by substituting the bound for  $\sigma(k)$  into the first theorem, and letting  $k$  be a prime number  $p$ , gives:

**Lemma 5** *There exist positive constants  $A_1, A_2, A_3, A_4$  such that for  $N, p, l$ , where  $p$  is prime,  $N$  is a natural number and  $l \in \{0, 1, \dots, p - 1\}$  we have:*

$$\pi(N, p, l) \geq \frac{1}{p-1} \int_2^N \frac{du}{\ln u} - \frac{A_1 N}{e^{A_3 \sqrt{\log N}}} - \frac{A_2 N}{N^{A_4 / (\sqrt{p} \log^2 p)} (p-1) \ln N}. \quad (1)$$

Lemma 4 easily follows from Lemma 5. For any  $\epsilon > 0$ , we can choose  $P(\epsilon)$  so that for  $p \geq P(\epsilon)$  and  $N \geq 2^p$  the first term on the right hand side of (1) is at least

$$(1 - \epsilon) \frac{N}{(p-1) \ln N} \geq (1 - \epsilon) \frac{N}{(\log N)^2},$$

and such that each of the other two terms is at most  $\epsilon N (\log N)^{-2}$ . Choosing  $\epsilon = 1/6$  gives Lemma 4.

### 3 Proof of the Main theorem

For the proof of our main result, we introduce one more number-theoretic function.

**Definition 1** For natural number  $j$ , let  $p_j$  denote the  $j^{\text{th}}$  smallest odd prime. The function  $\text{MULT}$  takes two integer arguments  $x$  and  $j$  where  $j$  is between 1 and  $\lceil \log |x| \rceil$ . and  $\text{MULT}(x, j) = \text{MOD}_{p_j}(x)$ , i.e., it is 1 if  $x$  is a multiple of  $p_j$  and is 0 otherwise.

Using Chinese remaindering and an observation of Boppa and Lagarias [BL87] we will show:

**Lemma 6**  $\text{MAJ} \leq_{\text{T}}^{\text{AC}^0} \text{MULT}$ .

Our main lemma is:

**Lemma 7**

1.  $\text{MULT} \leq_{\text{T}}^{\text{AC}^0} \text{GCD}$ .
2.  $\text{MULT} \leq_{\text{T}}^{\text{AC}^0} \text{PRIMES}$ .
3.  $\text{MULT} \leq_{\text{T}}^{\text{AC}^0} \text{SQUARE-FREE}$

Theorem 1 follows immediately by combining the two lemmas and the transitivity of  $\leq_{\text{T}}^{\text{AC}^0}$  reducibility.

### 3.1 Proof of Lemma 6

Fix  $n$  sufficiently large. We want to build a circuit to compute  $\text{MAJ}(x)$  for  $n$  bit strings  $x = (x_{n-1}, \dots, x_0)$  using the  $\text{MULT}$  function. By definition,  $\text{MAJ}(x) = \bigvee_{t=\lceil n/2 \rceil}^n \text{SUM}_t(x)$ , where  $\text{SUM}_t(x) = 1$  if  $x$  has exactly  $t$  1's. Thus it suffices for us to show how to compute  $\text{SUM}_t(x)$  for fixed  $t \leq n$ .

For integer  $s$  and natural number  $m$ , write  $s \pmod{m}$  for the unique integer  $r$  between 0 and  $m-1$  such that  $s-r$  is divisible by  $m$ . For natural number  $j$  and integer  $r$  satisfying  $0 \leq r < p_j$  define  $M_{j,r}(x)$  to be 1 if  $\sum_i x_i \pmod{p_j} = r$ .

Let  $k = \lceil \log n \rceil$ . By the prime number theorem,

$$p_k \sim k \ln k \sim \log n \ln \log n \leq n$$

(for  $n$  sufficiently large). For integer  $t$  and natural number  $j$ , let  $t_j = t \pmod{p_j}$ . The Chinese remainder theorem implies that since  $p_1 \cdots p_k \geq 2^k \geq n$ ,  $\sum_{i=1}^n x_i = t$  if and only if  $\sum_{i=1}^n x_i \pmod{p_j} = t_j$  for  $j \leq k$ . That is,

$$\text{SUM}_t(x) = \bigwedge_{j=1}^k M_{j,t_j}(x).$$

Thus it suffices to show that if  $j$  and  $r$  are integers satisfying  $1 \leq j \leq \lceil \log n \rceil$  and  $0 \leq r < p_j$  then  $M_{j,r}(x)$  can be computed (for  $n$ -bit strings  $x$ ) by a polynomial-size  $O(1)$ -depth circuit that uses  $\text{MULT}$  gates. This is a slight generalization of an observation of Boppana and Lagarias [BL87]. Since  $p_j$  is odd, there is some integer exponent  $u_j$ ,  $0 < u_j < p_j$ , such that  $2^{u_j} \equiv 1 \pmod{p_j}$ . For  $x = x_{n-1} \dots x_0$ , let  $f_j(x) = x_n 0^{u_j-1} x_{n-1} 0^{u_j-1} \dots 0^{u_j-1} x_1 0^{u_j-1} x_0$ . The integer represented in binary by  $f_j(x)$  is

$$f_j(x) = \sum_i 2^{u_j i} x_i.$$

Since

$$\sum_i x_i \equiv \sum_i x_i 2^{u_j i} \pmod{p_j}$$

we have that  $M_{j,r}(x) = \text{MULT}(f_j(x) - r, j)$ . This completes the proof of Lemma 6.

### 3.2 Proof of Lemma 7

We want to reduce the computation of  $\text{MULT}(x, j)$  (where by definition we need consider only the case where  $j \leq \lceil \log |x| \rceil$ , and hence  $p_j \leq |x|$  for all large  $x$ ) to each of the three given functions. We first note that by Proposition 3(1), we can build a bounded depth circuit of size  $n^{O(1)}$  that on input  $j$  outputs  $p_j$ .

The reduction of  $\text{MULT}$  to  $\text{GCD}$  is trivial, since it suffices to compute  $\text{GCD}(x, p_j, i)$  for all integers  $i \leq n$  to determine whether  $x$  is a multiple of  $p_j$ .

To reduce  $\text{MULT}$  to  $\text{PRIMES}$  we give a probabilistic test that uses  $\text{PRIMES}$  gates and tests whether an  $n$ -bit number  $x$  is a multiple of prime  $p_j \leq n$ . Given  $x$  and  $p_j$ , let  $w$  be the number of bits in  $p_j$  and choose a string of  $n-w+1$  bits at random and interpret the string as a random integer  $y$  in the range  $[-2^{n-w}, 2^{n-w} - 1]$ . The test rejects if  $x + p_j y$  is a prime different from  $p_j$  and accepts otherwise. If  $x$  is a multiple of  $p_j$ , the test will always accept. Suppose  $x$  is not a multiple of  $p_j$ . Then  $x + p_j y$  is uniformly distributed over all integers  $t$  that are congruent to  $x \pmod{p_j}$  and lie in some range  $[R_1, R_2]$  where  $-2^{n+1} < R_1 < 0$  and  $2^n \leq R_2 < 2^{n+1}$ . Lemma 4 implies that the test will reject with probability at least  $K/n^2$  for some constant  $K$ .

Now we construct a probabilistic circuit that performs  $Kn^3$  independent trials of this test in parallel and takes the AND of the trials. If  $x$  is a multiple of  $p_j$ , the circuit must accept. If  $x$  is not a multiple of  $p_j$ , then the probability that the circuit accepts is at most  $(1 - K/n^2)^{Kn^3} \leq e^{-n}$ . Now, as in the standard argument of [Adl78], there must be some setting of the random bits into this circuit that rejects any  $x$  that is not a multiple of  $p_j$ . Hardwiring the values of these bits yields a deterministic circuit  $C_j$  such that  $C_j(x)$  equals  $\text{MULT}(x, j)$ . Since there are only  $n^{O(1)}$  choices of  $j$ , we can combine the circuits  $C_j$  into one that on input  $(x, j)$ , uses  $j$  to select  $C_j$  and then computes  $C_j(x)$ . This circuit computes  $\text{MULT}(x, j)$ .

This construction is an  $\leq_{\text{T}}^{\text{AC}^0}$  reduction from  $\text{MULT}$  to  $\text{PRIMES}$ , since by Proposition 3 (parts 2 and 3),  $x + py$  can be computed by bounded depth circuits of size  $n^{O(1)}$ .

To reduce  $\text{MULT}$  to  $\text{SQUARE-FREE}$ , we need an analogue of Lemma 4 for square-free numbers. Accordingly for natural number  $n$ , prime  $p$  and integer  $l$  satisfying  $1 \leq l \leq p - 1$ , let  $S(N, p, l)$  denote the number of square-free numbers  $s$  satisfying  $0 \leq s \leq N - 1$ , such that  $s \equiv l \pmod{p^2}$ . To estimate  $S(2^n, p, l)$ , for natural number  $d$ , let  $T_d(N, p, l)$  be the number of integers  $s$ ,



$0 \leq s \leq N - 1$  such that

$$s \equiv l \pmod{p^2} \quad \text{and} \quad s \equiv 0 \pmod{d^2}.$$

By applying the inclusion-exclusion principle we derive that

$$S(2^n, p, l) = \sum_{1 \leq d \leq 2^{n/2}} \mu(d) T_d(2^n, p, l),$$

where  $\mu(d)$  is the Möbius function, which is defined to be 0 if  $d$  is not square-free, and otherwise is  $(-1)^{\nu(d)}$ , where  $\nu(d)$  is the number of prime divisors of  $d$ . Obviously,  $T_d(2^n, p, l) = 0$  if  $p$  divides  $d$  and

$$\left| T_d(2^n, p, l) - \frac{2^n}{p^2 d^2} \right| \leq 1$$

otherwise (because if  $\gcd(d, p) = 1$  the above system of congruences defines each such  $s \pmod{p^2 d^2}$  uniquely). Therefore

$$\left| S(2^n, p, l) - \sum_{\substack{1 \leq d \leq 2^{n/2} \\ \gcd(d, p) = 1}} \mu(d) \frac{2^n}{p^2 d^2} \right| \leq \sum_{\substack{1 \leq d \leq 2^{n/2} \\ \gcd(d, p) = 1}} |\mu(d)| \leq 2^{n/2}.$$

Furthermore,

$$\begin{aligned} & \left| \sum_{\substack{1 \leq d \leq 2^{n/2} \\ \gcd(d, p) = 1}} \mu(d) \frac{2^n}{p^2 d^2} - \frac{2^n}{p^2} \sum_{\substack{d=1 \\ \gcd(d, p) = 1}}^{\infty} \frac{\mu(d)}{d^2} \right| \\ & \leq \frac{2^n}{p^2} \left| \sum_{\substack{d > 2^{n/2} \\ \gcd(d, p) = 1}} \frac{\mu(d)}{d^2} \right| \leq \frac{2^n}{p^2} \sum_{\substack{d > 2^{n/2} \\ \gcd(d, p) = 1}} \frac{1}{d^2} \\ & \leq \frac{2^n}{p^2} \sum_{d > 2^{n/2}} \frac{1}{d(d-1)} = \frac{2^n}{p^2} \sum_{d > 2^{n/2}} \left( \frac{1}{d-1} - \frac{1}{d} \right) \\ & \leq \frac{2^n}{p^2 (2^{n/2} - 1)} < 2^{n/2}. \end{aligned}$$

Now, we have:

$$\begin{aligned} \sum_{\substack{d=1 \\ \gcd(d,p)=1}}^{\infty} \frac{\mu(d)}{d^2} &= \sum_{d=1}^{\infty} \frac{\mu(d)}{d^2} - \sum_{c=1}^{\infty} \frac{\mu(cp)}{(pc)^2} \\ &= \sum_{d=1}^{\infty} \frac{\mu(d)}{d^2} - \frac{1}{p^2} \sum_{\substack{c=1 \\ \gcd(c,p)=1}}^{\infty} \frac{-\mu(c)}{c^2}, \end{aligned}$$

which implies:

$$\sum_{\substack{d=1 \\ \gcd(d,p)=1}}^{\infty} \frac{\mu(d)}{d^2} = \frac{p^2}{p^2-1} \sum_{d=1}^{\infty} \frac{\mu(d)}{d^2}.$$

Now, it is known (see Theorem 4.4 of [Pr57]) that for  $s > 1$ ,

$$\sum_{d=1}^{\infty} \frac{\mu(d)}{d^s} = \frac{1}{\zeta(s)},$$

where  $\zeta(s) = \sum_{n=1}^{\infty} 1/n^s$  is the Riemann  $\zeta$ -function. Since  $\zeta(2) = \pi^2/6$ , we conclude:

$$|S(2^n, p, l) - \gamma(p)2^n| \leq 2^{n/2+1}, \quad (2)$$

where

$$\gamma(p) = \frac{6}{\pi^2(p^2-1)}.$$

Thus (2) provides the desired analogue of Lemma 4.

Now, to determine  $\text{MULT}(x, j)$  it is enough to check that  $xp_j$  is not divisible by  $p_j^2$ . Again let  $w$  be the binary length of  $p_j$ , choose a random string of  $n - w + 1$  bits and interpret the string as an integer  $y$  in the range  $[-2^{n-w}, 2^{n-w} + 1]$ . Using an oracle gate for SQUARE-FREE check if  $p_jx + p_j^2y$  is square-free. If  $x$  is a multiple of  $p_j$ , the oracle gate always rejects. If  $x$  is not a multiple of  $p_j$ , the oracle gate accepts with probability  $\gamma(p_j) + o(1)$ . The rest of the argument is analogous to the case of PRIMES.

## 4 Conclusions and Open Problems

The reduction of  $\text{MULT}(x, j)$  to PRIMES of the last section is *nonuniform* even for fixed  $j$ . That is, we can provide no efficient procedure to *build* the  $\text{AC}^0$  circuits that perform the reduction; we can show only that they exist,

via a probabilistic argument. Surely it is obvious that telling if a number is a multiple of 3 is no harder than telling if a number is composite! Is there a direct, *uniform* reduction that captures this intuition?

The theory of many-one reducibility has been extremely useful in characterizing the complexity of many problems, although it has not turned out to be very useful for studying number-theoretic problems. For example, although we know that  $\text{MOD}_3$  is  $\text{AC}^0$ -Turing reducible to  $\text{PRIMES}$ , we do not know if it is many-one reducible to  $\text{PRIMES}$ . Might it be possible to *prove* that there is no many-one reduction from  $\text{MOD}_3$  (or  $\text{PARITY}$ ) to  $\text{PRIMES}$ ? This would show that  $\text{PRIMES}$  is not complete for NP (under  $\leq_m^{\text{AC}^0}$  reductions), and in fact would show that it is not complete for any familiar complexity class. Although in general it is difficult to show that there *is* no  $\leq_m^{\text{AC}^0}$  reduction from one problem to another (since, for example, the  $\text{NP} \neq \text{NC}^1$  question can be phrased this way), it is worth noting that a set  $A$  in NP is presented in [AAIPR97] such that there is no  $\leq_m^{\text{AC}^0}$  reduction from  $\text{PARITY}$  to  $A$ .

If  $\text{PRIMES}$  were complete for NP (or for any other reasonable complexity class) under  $\leq_m^{\text{AC}^0}$  reductions, the isomorphism theorems of [AAR98, AAIPR97] show that  $\text{PRIMES}$  would be isomorphic to all of the other complete sets for that class, under isomorphisms computable and invertible by P-uniform depth-three  $\text{AC}^0$  circuits. In particular, there would be an isomorphism of this sort between  $\text{PRIMES}$  and  $\text{PRIMES} \times \{0, 1\}^*$ . Among other things, this would yield a fairly “dense” set of primes in P, by looking at the isomorphic image of  $\{2\} \times \{0, 1\}^*$ . (Observe that it was shown only fairly recently that there is an infinite set of primes in P [PPS89].) Perhaps the existence of such an isomorphism would bestow  $\text{PRIMES}$  with some properties that it provably does not have. Perhaps such an isomorphism must involve multiplication (which cannot be computed by  $\text{AC}^0$  circuits). That is, perhaps it is possible to prove that  $\text{PRIMES}$  is not complete for any familiar complexity class. Of course, in the foregoing discussion we are considering only *unconditional* proofs. It is well known, thanks to [Mil76], that  $\text{PRIMES}$  is in P under the Extended Riemann Hypothesis.

Additional observations and speculations of this sort pertaining to the factoring problem can be found in [All98].

We remark that  $\text{MULT}$  can be  $\text{AC}^0$ -reduced to other natural number-theoretic problems and thus these problems are also hard for  $\text{TC}^0$ . For example, consider the problem of computing the parity of  $\omega(x)$ , which is the

number of distinct prime divisors of  $x \in \mathbb{N}$ . For any prime  $p$ :

$$\text{MOD}_p(x) = 0 \iff \omega(x) + 1 \equiv \omega(px) \pmod{2}.$$

Thus MULT (and MAJ) is  $\leq_{\text{T}}^{\text{AC}^0}$  reducible to the parity of  $\omega$ .

It would be very interesting to obtain similar results for other number-theoretic problems. For example, it is shown [Shp99] that deciding quadratic residuosity modulo a large prime  $q$  is not in  $\text{AC}^0$ . Note that this question is equivalent to computing the rightmost bit of the discrete logarithm modulo  $q$ . It would be very desirable to extend this lower bound to the classes  $\text{AC}^0[p]$  and/or  $\text{TC}^0$ .

**Acknowledgment.** This paper was essentially written during a visit by the third author to Rutgers University, whose hospitality is gratefully acknowledged. We thank Lance Fortnow for pointing out that this version of our work proves that  $\text{MOD}_2$  is reducible to PRIMES, thus answering a question that was left open in the version that appeared in Proc. 14th Annual IEEE Conference on Computational Complexity, 1999. We thank Alan Woods for pointing out that [HS68] does not completely settle the question of whether primality can be determined in less than logarithmic space.

## References

- [Adl78] L. Adleman, *Two theorems on random polynomial time*, in “Proc. 19th IEEE Symposium on Foundations of Computer Science”, 1978, 75–83.
- [AH87] L. Adleman and M.-D. Huang, *Recognizing primes in random polynomial time*, in “Proc. 19th ACM Symposium on Theory of Computing”, 1987, 462–469.
- [APR83] L. Adleman, C. Pomerance and R. S. Rumely, *On distinguishing prime numbers from composite numbers*, *Annals Math.* **117** (1987), 173–206.
- [AAIPR97] M. Agrawal, E. Allender, R. Impagliazzo, T. Pitassi and S. Rudich, *Reducing the complexity of reductions*, in “Proc. 29th ACM Symposium on Theory of Computing”, 1997, 730–738.

- [AAR98] M. Agrawal, E. Allender and S. Rudich, *Reductions in circuit complexity: An isomorphism theorem and a gap theorem*, J. Comp. Sys. Sci. **57** (1998), 127–143.
- [All98] E. Allender, *News from the isomorphism front*, Computational Complexity Column, Bulletin of the EATCS **66** (1998), 73–82.
- [BDS98a] A. Bernasconi, C. Damm and I. E. Shparlinski, *Circuit and decision tree complexity of some number theoretic problems*, Tech. Report 98-21, Dept. of Math. and Comp. Sci., Univ. of Trier, 1998, 1–17.
- [BDS98b] A. Bernasconi, C. Damm and I. E. Shparlinski, *On the average sensitivity of testing square-free numbers*, in “Proc. 5th Intern. Computing and Combin. Conf.”, Lect. Notes in Comp. Sci., Springer-Verlag, Berlin, **1627** (1999), 291–299.
- [BDS99] A. Bernasconi, C. Damm and I. E. Shparlinski, *The average sensitivity of square-freeness*, Preprint, 1999, 1–14.
- [BS99] A. Bernasconi and I. E. Shparlinski, *Circuit complexity of testing square-free numbers*, in “Proc. 16th Intern. Symp. on Theor. Aspects in Comp. Sci.”, Lect. Notes in Comp. Sci., Springer-Verlag, Berlin, **1563** (1999), 47–56.
- [BL87] R. Boppana and J. Lagarias, *One-way functions and circuit complexity*, Information and Computation **74** (1987), 226–240.
- [CSV84] A.K. Chandra, L. Stockmeyer and U. Vishkin, *Constant depth reducibility*, SIAM J. Computing **13** (1984), 423–439.
- [HB76] J. Hartmanis and L. Berman, *On tape bounds for single letter alphabet language processing*, Theoretical Computer Science **3** (1976), 213–224.
- [HS68] J. Hartmanis and H. Shank, *On the recognition of primes by automata*, J ACM **15** (1968), 382–389.
- [LV99] R. Lipton and A. Viglas, *On the circuit complexity of primality*, manuscript.
- [Man92] S.-G. Mantzivis, *Circuits in bounded arithmetic, 1*, Ann. Math. Artificial Intelligence **6** (1992), 127–156.

- [Med91] J. Meidānis, *Lower bounds for arithmetic problems*, Inform. Proc. Letters **38** (1991), 83–87.
- [Mil76] G. Miller, *Riemann’s hypothesis and tests for primality*, J. Comput. System Sci. **13** (1976), 300–317.
- [Pa35] A. Page, *On the number of primes in an arithmetic progression*, Proc. Lond. Math. Soc. **39** (1935), 116–141.
- [PPS89] J. Pintz, W. Steiger and E. Szemerédi, *Two infinite sets of primes with fast primality tests*, Math. Comp. **53** (1989), 399–406.
- [Pr57] K. Prachar, Primzahlverteilung, Springer-Verlag, 1957.
- [R80] M. Rabin. *Probabilistic algorithm for primality testing*, Journal of Number Theory **12** (1980), 128–138.
- [Raz87] A. A. Razborov, *Lower bounds on the size of bounded depth networks over a complete basis with logical addition*, Matematicheskije Zametki, **41** (1987), 598–607, English translation in Mathematical Notes of the Academy of Sciences of the USSR **41** (1987), 333–338.
- [Shp98] I. E. Shparlinski, *On polynomial representations of Boolean functions related to some number theoretic problems*, Electronic Colloq. on Comp. Compl., TR98-054, 1998, 1–13.
- [Shp99] I. E. Shparlinski, Number theoretic methods in cryptography: Complexity lower bounds, Birkhäuser, 1999.
- [Smo87] R. Smolensky, *Algebraic methods in the theory of lower bounds for Boolean circuit complexity*, in “Proc. 19th ACM Symposium on Theory of Computing”, 1987, 77–82.
- [SS77] R. Solovay and V. Strassen, *A fast Monte Carlo test for primality*, SIAM J. Comput. **6** (1977), 84–85, erratum **7** (1978), 118.
- [Str] H. Straubing, *Finite Automata, Formal Logic, and Circuit Complexity*, Birkhäuser, 1994.