Research Article

Jacek Pomykała* and Maciej Radziejewski

Integer factoring and compositeness witnesses

https://doi.org/10.1515/jmc-2019-0023
Received Jul 09, 2019; accepted May 01, 2020

Abstract: We describe a reduction of the problem of factorization of integers $n \leq x$ in polynomial-time $(\log x)^{M+O(1)}$ to computing Euler's totient function, with exceptions of at most $x^{O(1/M)}$ composite integers that cannot be factored at all, and at most $x \exp\left(-\frac{c_M(\log\log x)^3}{(\log\log\log x)^2}\right)$ integers that cannot be factored completely. The problem of factoring square-free integers n is similarly reduced to that of computing a multiple D of $\phi(n)$, where $D \ll \exp((\log x)^{O(1)})$, with the exception of at most $x^{O(1/M)}$ integers that cannot be factored at all, in particular $O(x^{1/M})$ integers of the form $n = pq$ that cannot be factored.

Keywords: Large sieve, factoring algorithms, $Z_n^\star$-generating sets, Dirichlet characters, smooth numbers, Discrete Logarithm Problem for composite numbers, primality testing, Euler's totient function, RSA

2010 Mathematics Subject Classification: 11A05,11A15, 11A51,11M06,11Z05, Computational number theory: 11Y05,11Y16

1 Introduction

The computational problems of factorization of a composite integer n and computation of discrete logarithms in $\mathbb{Z}_n^\star$ play a significant role in the current public key cryptography. The security of many popular cryptosystems rests on the difficulty of the integer factorization problem. E.g., to break the RSA cryptosystem, it is enough to be able to factorize integers of the form $n = pq$. The reduction of the general factorization problem to computing the values of Euler's totient function $\phi(n)$ or to computing the discrete logarithms in $\mathbb{Z}_n^\star$ has attracted much attention in the last decades. The existence of such a reduction (which is trivial in the special case $n = pq$) would, of course, render the cryptosystems in question insecure if somebody developed a method to quickly compute, e.g., $\phi(n)$ for large n. Even if the computation of $\phi(n)$ seems out of reach at present, the computation of a multiple of $\phi(n)$ seems more plausible. Obviously $n!$ is one such multiple, but it is far too large for any practical purposes. If anyone came up with a fast method of computing such a multiple of reasonable size, then, as we show at the end of the paper, it would seriously impact the security of the RSA cryptosystem.

E. Bach (see [3]) showed the reduction of factoring n to solving the discrete logarithm problem in $\mathbb{Z}_n^\star$ in probabilistic polynomial time and, assuming the Extended Riemann Hypothesis (ERH), also in deterministic polynomial time. The corresponding unconditional deterministic subexponential time reduction was proved in [14]. Factoring integers given the values of Euler's totient function $\phi(n)$ can be done in probabilistic polynomial time due to the work [15]. The related deterministic polynomial time algorithm is known only under

The second author was supported by the Polish National Science Centre grant number 2017/25/B/ST1/00208

***Corresponding Author: Jacek Pomykała:** Faculty of Mathematics, Informatics and Mechanics, University of Warsaw ul. Banacha 2, PL-02-097 Warsaw, Poland; Email: pomykala@mimuw.edu.pl
Maciej Radziejewski: Faculty of Mathematics and Computer Science, Adam Mickiewicz University ul.Umultowska 87, PL-61-614 Poznań, Poland; Email: maciejr@amu.edu.pl

the assumption of ERH and was done in [12]. The unconditional result is still unsolved (see [1]). Although an unconditional, deterministic algorithm is known, it runs in subexponential time, cf. [18]. It was also noticed there, that almost all positive integers can be factorized in deterministic polynomial time with the oracle Φ, which is a hypothetical device for computing the value of $\phi(n)$ for any positive integer n (see the next section for the definitions of the oracles). Quite recently the analogous deterministic reductions were investigated in [6] and [13] with the oracles Φ and Dec Φ, the latter giving the complete prime factorization of $\phi(n)$.

The aim of this paper is to design algorithms, that make use of these oracles, with improved upper bounds for the number of integers that we are unable to factorize. We approach the solution of the problem posed in [1], showing that the related reduction in polynomial time $t = O(\log^{M+5} x)$ holds for all, but at most $O(x^{\frac{c}{M}})$ exceptions $n \le x$, where $c = 1.34$ and $M \ge 4$. The value of the constant c can be decreased to $c = 1$ in the case of the (stronger) oracle Dec Φ. This extends and improves substantially upon the related bound for the possible exceptions proved in [6].

We also investigate the problem of complete factorization with the aid of the oracle $O = \Phi$. In this case we were able to obtain the bound $O(x/(\log x)^{6.5M})$ for the number of the related exceptions when the oracle is queried once, and $x \exp\left(-\frac{c_M(\log\log x)^3}{(\log\log\log x)^2}\right)$, where $c_M \approx M^{3-\varepsilon}$, when the oracle is queried multiple times. The latter bound, while weaker than $x^{\frac{c}{M}}$, is also stronger than any bound of the form $x/(\log x)^c$. The former bound depends on the current top results related to the Vinogradov-Linnik problem on the least character non-residue. Any improvement upon the result quoted here as Lemma 3.4 will translate to an appropriate improvement of this bound.

In the last section we discuss similar reductions using slightly weaker oracles, related to the multiples of Euler's totient function. In the special case of integers of the form $n = pq$ we show that all except $O(x^{\frac{1}{M}})$ such integers $n \le x$ can be factored in time $t = O\big((\log x)^{M+O(1)}\big)$ with one query to an oracle returning a multiple D of $\phi(n)$ of size $\exp\big((\log x)^{O(1)}\big)$.

Our approach is based on the investigation of the corresponding "hard" numbers that may not be factored with the aid of the related Fermat-Euclid compositeness witnesses or power-difference compositeness witnesses of given order (see Section 2 for definition). We remark that in order for an algorithm to complete in polynomial time with probability 1, i.e. for almost all integers, it would be enough to know that there are no more than $o(x)$ hard integers in $[1, x]$, in other words the set of hard integers should have density zero. However, to say that a set has "density zero" is only a very rough estimate. For example, the set of primes, the set of integers of the form $n = pq$, the set of squares, and the set of cubes all have density zero, but the first two are still much "denser", with $O(\frac{x}{\log x})$ and $O(\frac{x\log\log x}{\log x})$ elements in $[1, x]$, respectively, than the other two, with $O(x^{1/2})$ and $O(x^{1/3})$ elements in $[1, x]$. The existence of a large set of hard integers might suggest that it might still be possible to keep the cryptosystem secure by appropriate choice of parameters. Having a tight estimate for the number of hard integers in $[1, x]$ makes such a measure unlikely. The additional arithmetical properties of hard integers stated in some lemmas and theorems serve the purpose of showing which integers might and which should not be considered as parameters in order to keep a cryptosystem secure in the hypothetical event of someone developing one of the oracles considered here.

We transform the problem to the investigation of primitive Dirichlet characters $\chi \bmod n$ of given order and prove that hard numbers correspond to the exceptional conductors of such characters. The estimates for the numbers of such conductors are deduced from the bounds for the least character nonresidue proved in [4] and [10], and the enhanced analysis of the Hensel-Berlekamp method applied in [18].

2 Notations and basic definitions

Conventionally m, n stand for positive integers while p, q are prime numbers, $\mathcal{A}$ a given deterministic algorithm, $\mathcal{O}$ — the related oracle. We also employ the following notations throughout the paper.

$\operatorname{lcm}(m, n)$ the least common multiple of m and n
$\gcd(m, n)$ the greatest common divisor of m and n

ϕ	Euler' totient function
$P^+(n)$, $P^-(n)$	the greatest and smallest prime divisors of n respectively
$\omega(n)$	the number of distinct prime divisors of n
$v_q(n)$	the exponent in the highest power of q dividing n
$\mathrm{ord}_n b$	the order of $b \bmod n$, where $\gcd(b, n) = 1$
$\log x$	the natural logarithm of x
$\mathrm{LN}(\chi)$	for a Dirichlet character $\chi(\bmod n)$, the least character nonresidue, i.e. the least b such that $\chi(b) \notin \{0, 1\}$
$F(x, \mathcal{A}, \mathcal{O}, t_{\mathcal{A}}, t_{\mathcal{O}})$	the number of positive integers $n \le x$ that can be factored completely by algorithm $\mathcal{A}$ in time $t_{\mathcal{A}}$ with at most $t_{\mathcal{O}}$ queries to oracle $\mathcal{O}$; at each oracle query the input is a positive integer not exceeding n
$F^\star(x, \mathcal{A}, \mathcal{O}, t_{\mathcal{A}}, t_{\mathcal{O}})$	the number of positive integers $n \le x$ that either are prime, or can be nontrivially factored by algorithm $\mathcal{A}$ in time $t_{\mathcal{A}}$ with at most $t_{\mathcal{O}}$ queries to oracle O; at each oracle query the input is a positive integer not exceeding n
Φ	an oracle that, given a positive integer n, always returns the value of $\phi(n)$
$\mathrm{Dec}\,\Phi$	an oracle that, given a positive integer n, always returns the complete prime factorization of $\phi(n)$
$\mathrm{Mul}^{M'}\,\Phi$	an oracle that, given a positive integer n, always returns a multiple D of $\phi(n)$ of size at most $D = O(\exp((\log n)^{M'}))$; here M' is an arbitrary fixed positive parameter
$\mathrm{Dec}\,\mathrm{Mul}^{M'}\,\Phi$	an oracle that, given a positive integer n, always returns the prime factorization of a multiple D of $\phi(n)$ of size at most $D = O(\exp((\log n)^{M'}))$; here M' is an arbitrary fixed positive parameter.

We note that our algorithms, and thus the functions $F(x, \mathcal{A}, \mathcal{O}, t_{\mathcal{A}}, t_{\mathcal{O}})$ and $F^\star(x, \mathcal{A}, \mathcal{O}, t_{\mathcal{A}}, t_{\mathcal{O}})$, implicitly depend on the choice of some auxiliary parameters, denoted as B, y and z. We explicitly mention these parameters in Theorems 3.1, 3.5, 4.1, 5.1 and 6.1 stating lower bounds for F and $F^\star$. Optimized values for these parameters, all of the order $(\log x)^{O(1)}$, may be found in the proofs of the theorems.

The investigated factoring algorithms are based on two kinds of factorization witnesses. The first is the so called Fermat-Euclid compositeness witness, i.e. an element b such that

$$\gcd\left(b^{\frac{\mathrm{ord}_n b}{r}} - 1, n\right) \neq 1. \tag{1}$$

for some prime $r \mid \mathrm{ord}_n b$ (more precisely it is called Fermat-Euclid compositeness witness of order r for n). For $r = 2$ we call such b a Miller-Rabin compositeness witness. Let $r \mid \phi(n)$ be a prime, $l = v_r(\phi(n))$, $k \in \{1, \ldots, l\}$, $u = \phi(n) r^{-l+k-1}$, and $b_0 = \min\{b \ge 1 : v_r(\mathrm{ord}_n b) = k\}$. We call b a power-difference compositeness witness of order r and degree k if $v_r(\mathrm{ord}_n b) = k$ and for some $j = 1, \ldots, r-1$ we have

$$1 < \gcd(b^u - b_0^{uj}, n) < n.$$

This notion, admittedly harder to employ than that of a Fermat-Euclid witness, is useful for small primes $r \ge 3$, for which the iteration over j is possible. The idea is that if b and b_0 are not Fermat-Euclid witnesses of order r, then for each $p \mid n$ both b^u and b_0^u are of order $r \bmod p$, so one is the j-th power of the other $(\bmod p)$ for some $j = 1, \ldots, r-1$. Unless j is the same for every p, the gcd will yield a nontrivial factorization of n. In case of $r = 2$ this would not be useful, because the only element of order 2 mod p is -1, so if b is a power-difference witness for a square-free n (or indeed any n not divisible by 8), then b or b_0 is a Miller-Rabin witness for n.

A number n is called $(\mathcal{A}, B, y)$-hard if $P^+(n) > y$ and the algorithm $\mathcal{A}$ does not find the complete factorization of n with the aid of Fermat-Euclid compositeness witnesses $b \le B$. The number n is called $(\mathcal{A}, B, y)^\star$-hard if $P^-(n) > y$ and the algorithm $\mathcal{A}$ does not find any nontrivial divisor of n with the aid of Fermat-Euclid compositeness witnesses $b \le B$.

In the following sections we present several algorithms factoring square-free positive integers. This limitation would affect only a thin set of integers, because we always factor out small primes by brute force. By [17, Theorem I.4.2] the number of positive integers $n \le x$ with all prime factors $p > y$ is of the order

$$x \prod_{p \le y} (1 - 1/p) \asymp x / \log y,$$

assuming $y \le x^{1/\log\log(x)}$ (in our case y is in fact much smaller). The number of such n divisible by a square of a prime is at most

$$\int_{y}^{x^{1/2}y^{-\log\log(x)/2}} \frac{x}{t^2\log y}\,d\pi(t) + \int_{x^{1/2}y^{-\log\log(x)/2}}^{\sqrt{x}} \frac{x}{t^2}\,d\pi(t) \ll$$

$$\ll \frac{\sqrt{x}}{\log x} + \frac{x}{\log y}\int_{y}^{x^{1/2}y^{-\log\log(x)/2}} t^{-3}\pi(t)\,dt + x\int_{x^{1/2}y^{-\log\log(x)/2}}^{\sqrt{x}} t^{-3}\pi(t)\,dt \ll \frac{x}{y\log^2 y},$$

provided $y \le x^{1/2\log\log(x)}$. However, our sets of hard numbers will be much thinner than this, so it is useful to deal with non-square-free numbers. Given an oracle $O = \Phi$ we can easily reduce the general problem of factoring an integer to the square-free case via the familiar algorithm of Landau [9], of complexity $O(\log^3 n)$, that we denote by $\mathcal{A}_0$. With $\mathcal{A}_0$ every positive integer n can be represented as the product $n = n_1 n_2^2 \ldots n_s^s$ of powers of pairwise coprime, square-free numbers n_i, using $O(\omega(n))$ calls to the Φ oracle. Thus it reduces the problem of factoring n to the problem of factoring the square-free numbers n_i, $i \le s$. The complexity $O(\log^3 n)$ will be negligible when compared with the factoring of square-free factors $n_i \mid n$. If $\mathcal{A}$ is a factorization algorithm for square-free numbers, we consider a composite algorithm using both $\mathcal{A}$ and $\mathcal{A}_0$ to factorize general numbers. In the composite algorithm we run $\mathcal{A}_0$ for a given n and, inside it, we immediately let $\mathcal{A}$ factorize every new square-free factor $m \mid n$ found by $\mathcal{A}_0$. This way we can factor out the prime factors of m from n and update the value of $\phi(n)$ without a new query to the oracle. We can therefore reduce the number of oracle queries to 1. We denote the resulting composite algorithm by $\mathcal{A}_0(\mathcal{A})$. We also append any required extra parameters to this notation, or skip them if they have been fixed. E.g., we define an algorithm $\mathcal{A}_3$ that requires parameters B and y, and then we refer to $\mathcal{A}^\star$-hard numbers where $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_1), B, y)$. For such algorithms we evaluate the related failure sets by careful use of estimates related to Dirichlet's characters.

3 Factoring based on witnesses of small order

Algorithm 1 ($\mathcal{A}_1$) Miller-Rabin type complete factoring with the Φ oracle.

Input Square-free positive integer n, a positive multiple D of $\phi(n)$, auxiliary parameters B, y, where $B \le y$.
Output Factorization of n.

1. Factor out small prime divisors $p \mid n$, $p \le y$, using division with remainder.
2. For each $b = 1, \ldots, B$ find the smallest i such that $n \mid b^{D'2^i} - 1$, where $D' = D/2^{v_2(D)}$. Compute $d = \gcd(b^{D'2^{i-1}} - 1, n)$.
3. If any d is a nontrivial factor of n, return to Step 2 with $n' \in \{d, n/d\}$. Otherwise output n.

Theorem 3.1. *We have, for arbitrary fixed $M \ge 4$, $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_1), B, y)$, and appropriate choices of B and y:*

$$F(x, \mathcal{A}, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x(\log x)^{-6.5M}\right)$$

and

$$F^\star(x, \mathcal{A}, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x^{1.34/M}\right),$$

where $t_{\mathcal{O}} = 1$ and $t_{\mathcal{A}} = O((\log x)^{M+5})$.

The proof of Theorem 3.1 is based on the estimates related to (exceptional) characters χ for which the least character nonresidue $\mathrm{LN}(\chi)$ is large. We call an integer n a B-exceptional integer if there exists a primitive Dirichlet character $\chi \bmod n$ such that $\mathrm{LN}(\chi) > B$.

Lemma 3.2. *Let n be composite, odd, square-free, let $r \mid \phi(n)$ be a prime, and let*

$$k = \max_{b \le B} v_r(\mathrm{ord}_n b).$$

Suppose that all prime divisors of n are greater than B and that there is no Fermat-Euclid factorization witness $b \le B$ of order r for n. Then at least one of the following assertions holds:

- (*i*) *there exists a primitive Dirichlet character $\chi \pmod n$, whose order is a power of r, such that $\mathrm{LN}(\chi) > B$,*
- (*ii*) *we have $r = 2$, $2 \nmid \omega(n)$, and there exist numbers n_1, n_2 and primitive characters $\chi_1 \pmod{n_1}$ and $\chi_2 \pmod{n_2}$, whose orders are powers of 2, such that $\mathrm{lcm}(n_1, n_2) = n$, $n_1 < n^{2/\omega(n)}$, $n_1 n_2 < n^{1+1/\omega(n)}$, and $\mathrm{LN}(\chi_i) > B$ for $i = 1, 2$,*
- (*iii*) *we have $r \ge 3$ and there is a power-difference factorization witness $b \le B$ of order r and degree k for n, or*
- (*iv*) *we have $r \ge 3$ and $k = 0$.*

Proof. Let $p, q \mid n$, $r \mid p - 1$. We have $v_r(\mathrm{ord}_q b) = v_r(\mathrm{ord}_p b) = v_r(\mathrm{ord}_n b)$ for all $b \le B$. Suppose, as we may, that assertions (*iii*) and (*iv*) are false. Then, if $k = 0$, we would have $r = 2$ and every quadratic character $\chi \pmod n$ would satisfy (*i*). We can therefore suppose $k > 0$.

We have $r \mid q - 1$, because $r^k \mid q - 1$. Let $u = \phi(n) r^{-v_r(\phi(n))+k-1}$. Let χ be any character of order $r^{v_r(p-1)-k+1} \bmod p$, i.e. the u-th power of any character of order $p - 1 \bmod p$. Then for each $b \le B$ we have $\mathrm{ord}\chi(b) = r$ if $v_r(\mathrm{ord}_p(b)) = k$, and $\chi(b) = 1$ otherwise. Moreover there exists a smallest $b_0 \le B$ such that $\mathrm{ord}(\chi(b_0)) = r$. Let ψ be any character of order $r^{v_r(q-1)-k+1} \bmod q$, i.e. the u-th power of any character of order $q - 1 \bmod q$. Then

$$\mathrm{ord}\chi(b) = \mathrm{ord}\psi(b) \in \{1, r\}, \qquad \text{for all } b \le B.$$

For every $b \le B$ let $m(b) \in \{0, 1, \dots, r - 1\}$ be such that $\chi(b) = \chi(b_0)^{m(b)}$, and let $l \in \{1, \dots, r - 1\}$ be such that $\psi(b_0) = (\chi(b_0))^l$.

Suppose $\psi(b) \ne \psi(b_0)^{m(b)}$ for some $b \le B$. Then $b_0^{um(b)}$ and b^u would be congruent mod p and not mod q, so b would be a power-difference witness of order r and degree k, contrary to our assumption. Hence we have $\psi(b) = \psi(b_0)^{m(b)} = (\chi(b_0))^{lm(b)} = \chi(b)^l$ for all $b \le B$. Therefore the character $\psi\chi^{-l} \bmod pq$ is equal to 1 on all $b \le B$. It is primitive, as a product of primitive characters to relatively prime moduli.

If p, q_1 and q_2 are any three primes dividing n, and we define χ, ψ_1, ψ_2 in an analogous way as χ and ψ before, then the characters $\psi_1\chi^{-l_1} \bmod pq_1$ and $\psi_2\chi^{-l_2} \bmod pq_2$ will be equal to 1 on all $b \le B$. Then at least one of the characters $\psi_1\psi_2\chi^{-l_1-l_2} \bmod pq_1q_2$ or $\psi_1\psi_2^{-1}\chi^{-l_1+l_2} \bmod pq_1q_2$ will be primitive (and equal to 1 on all $b \le B$), unless $l_1 + l_2 = r$ and $l_1 - l_2 = 0$, implying $r = 2$.

In the case $r \ge 3$ we can therefore split the set of prime divisors of n to groups of 2 or 3 factors, obtain the necessary primitive character mod the product of the factors in each group, and multiply them to obtain the primitive character mod n required in (*i*). When $r = 2$ and $2 \mid \omega(n)$, we can just use groups of two factors and show (*i*) again.

Finally, when $r = 2$ and $n = p_1 p_2 \dots p_{2m+1}$, $p_1 < \dots < p_{2m+1}$, we find the necessary primitive characters mod p_1p_2, mod p_1p_3, and mod p_ip_{i+1} for $i = 4, 6, \dots 2m$. Taking $n_1 = p_1p_2$ and $n_2 = p_1p_3 \dots p_{2m+1}$ and multiplying the characters mod p_1p_3 and mod p_ip_{i+1} for $i = 4, 6, \dots 2m$ we obtain the characters required in (*ii*). The assertion $n_1 < n^{2/\omega(n)}$ follows from $(\log p_1 + \log p_2)/2 < (\log p_1 + \dots + \log p_{2m+1})/(2m + 1)$. □

Lemma 3.3 (Baier [4, cf. Theorem 1.1])**.** *For every $a \ge 5/2$, $\varepsilon > 0$ we have $\mathrm{LN}(\chi) \le (\log x)^a$ for all but $O(x^{1/(a-3/4)+\varepsilon})$ primitive Dirichlet characters with conductor $q \le x$.*

Proof. The original theorem by S. Baier implies in particular that the number of exceptions is bounded by $O(x^{\max(f_1(a,2),f_2(a,2))+\varepsilon})$, where

$$f_1(a, 2) = \frac{1}{a - 3/4}, \qquad f_2(a, 2) = \frac{4}{3} \cdot (1 - 2(a - 2)f_1(a, 2)).$$

In the case $a \geq 5/2$ we have $f_1(a, 2) \geq f_2(a, 2)$. □

Lemma 3.4 (Lau, Wu [10, cf. Theorem 1]). *For every non-principal Dirichlet character $\chi \pmod q$, where q is cube-free, we have* $\mathrm{LN}(\chi) \ll q^{2/13}$.

Proof of Theorem 3.1. Let $B = (\log x)^a$, $a \geq 5$, and let $\varepsilon > 0$ be small with respect to a. Let $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_1), B, y)$. Suppose $\mathcal{A}$ is run for some $n \leq x$. The number k of final factors is $\ll \log x$. They are obtained in $\leq k - 1$ factorization steps where in each step one intermediate factor is replaced by two (intermediate or final) factors. So the total number of executions of Step 2 of $\mathcal{A}_1$ is $\ll \log x$. Each execution of Step 2 takes $\ll (\log D)B$ multiplications and $\ll (\log D)B$ gcd computations. As a result, Step 2 contributes $\ll (\log x)^{a+3+\varepsilon}$ in total to the overall complexity. The complexity of generating primes $\leq y$ and factoring them out is $\ll y(\log x)^{1+\varepsilon}$. We put $y = (\log x)^{a+2}$. Having in mind the complexity of algorithm $\mathcal{A}_0$ the complexity of algorithm $\mathcal{A}$ is

$$\ll (\log x)^{a+3+\varepsilon} + \log^3 n \ll (\log x)^{a+3+\varepsilon}.$$

Let S denote the set of $\mathcal{A}^*$-hard integers, E the set of B-exceptional integers, $S(x)$ and $E(x)$ the corresponding counting functions. Every $n \in S$ is square-free. By Lemma 3.2 every $n \in S$, $n \leq x$, is either exceptional itself or it is determined by a pair of two exceptional integers, $n_1 \leq x^{2/3}$ and $n_2 < \min(x, x^{4/3}/n_1)$ (since $2 \nmid \omega(n)$ implies $\omega(n) \geq 3$). Therefore

$$S(x) \leq E(x) + \sum_{\substack{n_1, n_2 \in E \\ n_1 \leq x^{1/3} \\ n_2 < x}} 1 + \sum_{\substack{n_1, n_2 \in E \\ x^{1/3} < n_1 < x^{2/3} \\ n_2 < x^{4/3}/n_1}} 1 \leq E(x) + E(x^{1/3})E(x) + \int_{x^{1/3}}^{x^{2/3}} E(x^{4/3}/t)\, dE(t)$$

It follows from Lemma 3.3 that

$$\begin{aligned} S(x) &\ll_a x^{(4/3)(1/(a-3/4)+3\varepsilon/5)} + \int_{x^{1/3}}^{x^{2/3}} E(x^{4/3}/t)\, dE(t) \\ &\ll x^{(4/3)(1/(a-3/4)+3\varepsilon/5)} + x^{(4/3)(1/(a-3/4)+3\varepsilon/5)} \int_{x^{1/3}}^{x^{2/3}} t^{-1/(a-3/4)-3\varepsilon/5}\, dE(t) \\ &\ll x^{(4/3)(1/(a-3/4)+3\varepsilon/5)} \left(1 + \int_{x^{1/3}}^{x^{2/3}} t^{-1-1/(a-3/4)-3\varepsilon/5} E(t)\, dt\right) \\ &\ll x^{(4/3)(1/(a-3/4)+3\varepsilon/5)} (\log x) \ll_a x^{4/(3a-9/4)+\varepsilon}. \end{aligned}$$

The smallest element n_0 of E satisfies $n_0 \gg B^{13/2} = (\log x)^{13a/2}$ by Lemma 3.4. Lemma 3.2 shows that every $\mathcal{A}$-hard integer is a multiple of a B-exceptional one. The number of $\mathcal{A}$-hard integers $n \leq x$ is thus at most

$$\begin{aligned} \int_{n_0^-}^{x} \frac{x}{t}\, dE(t) &\leq E(x) + x \int_{n_0^-}^{x} \frac{E(t)}{t^2}\, dt \ll_a x^{4/(3a-9/4)+\varepsilon} + x \int_{n_0}^{x} t^{-2+4/(3a-9/4)+\varepsilon}\, dt \ll x n_0^{-1+4/(3a-9/4)+\varepsilon} \\ &\ll x(\log x)^{-13a/2+26a/(3a-9/4)+13\varepsilon/2} \leq x(\log x)^{-13a/2+11}. \end{aligned}$$

This completes the proof of Theorem 3.1 with $M = a - 7/4$. □

Algorithm 2 ($\mathcal{A}_2$) Factorization based on witnesses of small orders and the Φ oracle

Input Square-free positive integer n, a positive multiple D of $\phi(n)$, auxiliary parameters B, y, z where $B \le y$.
Output Factorization of n.

1. Factor out small prime divisors $p \mid n$, $p \le y$, using division with remainder.
2. For each prime $r \mid D$, $r \le z$, perform steps 3–4.
3. For each $b = 1, \dots, B$ find the smallest $i = i(b)$ such that $n \mid b^{D' r^i} - 1$, where $D' = D/r^{v_r(D)}$. Compute $d = \gcd(b^{D' r^{i-1}} - 1, n)$.
4. Let $k = \max_{b \le B} i(b)$ and $b_0 = \min_{i(b)=k} b$. If $r \ge 3$ and $k \ge 1$, then for each $b = b_0 + 1, \dots, B$ and $j = 1, \dots r-1$ compute $d = \gcd(b^{D' r^{k-1}} - b_0^{D' r^{k-1} j}, n)$.
5. If any d is a nontrivial factor of n, return to Step 2 with $n' \in \{d, n/d\}$. Otherwise output n.

Algorithm $\mathcal{A}_2$ exploits the notion of power-difference witnesses. It is a generalization of $\mathcal{A}_1$, as it reduces to $\mathcal{A}_1$ when $z = 2$. Accordingly, Theorem 3.5 generalizes Theorem 3.1. The sets of hard and *-hard numbers for $\mathcal{A}_2$ are contained in the corresponding sets for $\mathcal{A}_1$. Conjecturally (assuming ERH) all of these sets are empty in the range of parameters given, as $\mathrm{LN}(\chi) \ll (\log q)^2$ by the result of N. C. Ankeny [2], so we do not aim to prove proper inclusion, however, better unconditional bounds would be desirable. In the present paper we are only able to state additional arithmetical consequences of $\mathcal{A}_2$-hardness, without improving upon the bounds. Nevertheless we include this algorithm, because power-difference witnesses potentially offer a new line of attack on the factorization problem.

Theorem 3.5. *We have, for arbitrary fixed $M \ge 4$, $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_2), B, y, z)$, appropriate choices of B and y, and arbitrary $z \ge 2$:*

$$F(x, \mathcal{A}, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x \left(1 - O_M((\log x)^{-6.5M})\right)$$

and

$$F^\star(x, \mathcal{A}, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x^{1.34/M}\right),$$

where $t_{\mathcal{O}} = 1$ and $t_{\mathcal{A}} = O((\log x)^{M+5} z)$. If $n \le x$ is $\mathcal{A}^\star$-hard, then for every odd prime $r \le z$ such that $r \mid \phi(n)$ there exists a primitive Dirichlet character $\chi \bmod \prod_{p \mid n, r \mid p-1} p$, whose order is a power of r, such that $\mathrm{LN}(\chi) > B$.

Proof. Let $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_2), B, y, z)$, B, ε as in the proof of Theorem 3.1, $z \ge 2$. The number of times to enter Step 2 of $\mathcal{A}_2$ is again $\ll \log x$, and each time the search for small primes takes $\ll z(\log x)^{1+\varepsilon}$. The total time spent in Step 2 itself is therefore $\ll z(\log x)^{2+\varepsilon}$. Each time Step 2 is executed, there are $\ll \min(z, \log x)$ values of r selected, for which Steps 3–4 are performed. Each execution of Step 3 takes $\ll (\log D)B$ multiplications and $\ll (\log D)B$ gcd computations. Step 3 thus contributes $\ll (\log x)^{a+3+\varepsilon} z$ to the overall complexity. Step 4 takes $\ll (z + \log D)B$ multiplications and $\ll zB$ gcd computations in each execution. The total contribution of Step 4 is

$$\ll (\log x)^{a+2+\varepsilon} \min(z, \log x) \max(z, \log x) = (\log x)^{a+3+\varepsilon} z.$$

The estimates in the assertion follow from Theorem 3.1. The additional property of $\mathcal{A}^\star$-hard numbers follows from Lemma 3.2 (in the case of $k = 0$ the exceptional character is any character of order r). □

4 Fermat-Euclid compositeness witnesses and nontrivial factorization

Theorem 4.1. *We have, for arbitrary fixed $M \ge 2$, $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_3), B, y)$, and appropriate choices of B and y:*

$$F(x, \mathcal{A}, \mathrm{Dec}\, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x \exp\left(-\frac{M^3 (\log\log x)^3}{9(\log(M+2) + \log\log\log x)^2}\right)\right)$$

Algorithm 3 ($\mathcal{A}_3$) Factorization based on the Dec Φ oracle

Input Square-free positive integer n, the prime factorization of $\phi(n)$, auxiliary parameters B, y, where $B \le y$.
Output Factorization of n.

1. Factor out small prime divisors $p \mid n$, $p \le y$, using division with remainder.
2. For each $1 \le b \le B$ use the original prime factorization of $\phi(n)$ (algorithm input) to compute $\operatorname{ord}_n b$ and the exponent $m = \operatorname{lcm}_{b \le B} \operatorname{ord}_n b$. (Exponent computing)
3. For each $1 \le b \le B$, and all primes $r \mid \operatorname{ord}_n b$, compute $d = \gcd(b^{\operatorname{ord}_n b/r} - 1, n)$. If any d is a nontrivial factor of n, return to Step 2 with $n' \in \{d, n/d\}$. (FE testing)
4. For all $l \le w$, $w = (\log n)^2 B$, compute $d = \gcd(lm + 1, n)$. If any d is a nontrivial factor of n, return to Step 2 with $n' \in \{d, n/d\}$. (w-distance prime detection)
5. Represent n in base m, i.e. $n = 1 + a_1 m + \cdots + a_k m^k$, $0 \le a_i < m$. Attempt to factorize $g(X) = 1 + a_1 X + \ldots + a_k X^k = \prod_{i \le k}(1 + b_i X)$ in $\mathbb{Z}[X]$ using the Hensel-Berlekamp method, as in the proof of [18, Lemma 8.5], cf. Lemma 4.3. If successful, output the factorization $n = p_1 \ldots p_k$, where $p_i = 1 + b_i m$, $i = 1, \ldots, k$. Otherwise output n. (Hensel-Berlekamp method)

and

$$F^\star(x, \mathcal{A}, \operatorname{Dec}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x^{1/M}\right),$$

where $t_{\mathcal{O}} = 1$ *and* $t_{\mathcal{A}} = O((\log x)^{M+5})$.

For $x, y > 0$ let $\psi(x, y)$ denote the number of integers $1 \le n \le x$ with $P^+(n) \le y$, so-called y-smooth integers.

Lemma 4.2 (Konyagin, Pomerance [8, Theorem 1])**.** *If* $x \ge 4$, $2 \le y \le x$, *then* $\psi(x, y) > x^{1 - \log\log x/\log y}$.

Lemma 4.3 (cf. Źrałek [18, Lemma 8.5])**.** *Let* $n = p_1 p_2 \ldots p_s$ *be composed of distinct odd primes* p_i. *Assume that* m *divides* $p_i - 1$, $p_i - 1 = m b_i$, *and* $b_i > s$ *for* $i = 1, 2, \ldots, s$, *and* $m^{s+1} \ge n$. *Then the complete factorization of* n *can be obtained with the Hensel-Berlekamp algorithm in* $O\left(\log^5 n (\log\log n)^2\right)$ *deterministic time.*

Proof. The original lemma of Źrałek contains a slightly stronger assumption $m^{s+1} > n\binom{s}{\lfloor s/2 \rfloor}$, and it does not include the (harmless) assumption $b_i > s$. The idea is to make sure that the coefficients of the polynomial product

$$g(X) = 1 + a_1 X + \ldots + a_s X^s = \prod_{i \le s}(1 + b_i X)$$

satisfy $a_i < m$ for $i = 1, \ldots, s$, so they can be determined by expressing $n = g(m)$ in base m. Suppose, as we may, that $b_1 < b_2 < \ldots < b_s$. We observe that for each $i = 1, \ldots, s$ we have

$$a_i \le \binom{s}{i} b_{s-i+1} b_{s-i+2} \ldots b_s < \binom{s}{i+1} b_{s-i} b_{s-i+1} \ldots b_s < \ldots < b_1 b_2 \ldots b_s,$$

where the first strict inequality follows from $b_{s-i} > s$. Hence $a_i < n/m^s \le m$. □

Lemma 4.4. *Suppose a positive, odd, square-free integer* n *is* $(\mathcal{A}_3, B, y)^\star$*-hard, with* $B = (\log n)^\eta$, $\eta > 3$, $w \ge (\log n)^{\eta+2}$, *and* $y \ge B$. *Let* $n = p_1 \cdot \ldots \cdot p_s$ *be the prime factorization of* n, *with* $p_1 < \ldots < p_s$. *Then we have*

$$\eta - 1 < s < \frac{\log n}{\eta \log w}, \tag{2}$$

$$p_s < n^{\eta/((\eta-1)(s+1))} < n^{1/(\eta-1)}, \tag{3}$$

and

$$n > \exp\left(\frac{1}{9}(\log B)^3 (\log\log B)^{-2}\right). \tag{4}$$

Proof. For every prime $p_i \mid n$ and every $b \le B$ we have $\mathrm{ord}_{p_i} b = \mathrm{ord}_n b$, otherwise a factorization witness would be found. Therefore the exponent m of the subgroup generated by positive integers $b \le B$ is the same in $\mathbb{Z}_n^\star$ and in each $\mathbb{Z}_{p_i}^\star$. Each B-smooth integer less than p_s belongs to the subgroup in $\mathbb{Z}_{p_s}^\star$, and for cyclic groups the exponent is equal to the group order, hence

$$m \ge \psi(p_s, B) > p_s^{1-1/\eta}, \tag{5}$$

where the second inequality follows from Lemma 4.2. Since $\psi(p_s, B) > \psi(\sqrt[s]{n}, B)$, we also have

$$m > \psi(\sqrt[s]{n}, B) > n^{1/s - \log\log n/(s \log B)} \tag{6}$$

by Lemma 4.2 again. The exponent of a subgroup divides the group order, so

$$m \mid \gcd(p_1 - 1, \ldots, p_s - 1).$$

Moreover

$$mw < p_1, \tag{7}$$

hence $s < \log n / \log y < w < \frac{p_1 - 1}{m}$, so we can apply Lemma 4.3 and obtain

$$m^{s+1} < n. \tag{8}$$

Further we have

$$(1 - \frac{1}{\eta}) \log p_s < \frac{1}{s+1} \log n$$

by (5) and (8). This implies the first inequality in (3) and also, by $(\log n)/s < \log p_s$, shows

$$\eta < s + 1,$$

i.e. the first inequality in (2) and the second one in (3). By (6) and (7) we obtain

$$w n^{1/s - \log\log n/(s \log B)} < n^{1/s},$$

which reduces to

$$s \log w < \log n \frac{\log\log n}{\log B} = \frac{\log n}{\eta},$$

implying the second inequality in (2). Furthermore (2) implies

$$\log n > (\eta - 1)\eta \log w \ge (\eta - 1)\eta(\eta + 2) \log\log n > \eta^3 \log\log n = (\log B)^3 (\log\log n)^{-2}. \tag{9}$$

Now suppose (4) is false, i.e.

$$\log n \le \frac{1}{9} (\log B)^3 (\log\log B)^{-2}. \tag{10}$$

Then we have

$$\log\log n \le 3 \log\log B - 2 \log\log\log B < 3 \log\log B, \tag{11}$$

where the last inequality comes from

$$\log\log\log B = \log\log(\eta \log\log n) > \log\log(3 \log\log 15) > 0.$$

However (11) and (9) together contradict (10). □

Proof of Theorem 4.1. Let x be large, $B = (\log x)^a$, $a > 3$, $y = B$ and $w = (\log x)^{a+2}$, and let ε be small, given a. Let $\mathcal{A} = (\mathcal{A}_0(\mathcal{A}_3), B, y)$. Let S denote the set of $\mathcal{A}^\star$-hard integers, E the set of B-exceptional integers, $S(x)$ and $E(x)$ the corresponding counting functions. It follows from Lemma 3.2, similarly to the proof of Theorem 3.1, that every $n \in S$, $n \le x$, is either B-exceptional itself or it is determined by a pair of two B-exceptional integers, $n_1 \le x^{2/\omega(n)}$ and $n_2 < \min(x, x^{1+1/\omega(n)}/n_1)$. It follows from Lemma 4.4 that

$$\omega(n) > \frac{\log B}{\log\log n} - 1 \ge a - 1.$$

We obtain, using Lemma 3.3 as before,

$$S(x) \le E(x^{1/(a-1)})E(x) + \int_{x^{1/(a-1)}}^{x^{2/(a-1)}} E(x^{1+1/(a-1)}/t)\, dE(t)$$
$$\ll x^{(1+1/(a-1))(1/(a-3/4)+\varepsilon/2)}(\log x) \ll_a x^{a/((a-1)(a-3/4))+\varepsilon} \ll x^{1/(a-7/4)}.$$

Every $\mathcal{A}$-hard integer is a multiple of an element of S. By Lemma 4.4 the smallest element n_0 of S satisfies the inequality

$$n_0 > \exp\left(\frac{1}{9}(\log B)^3(\log\log B)^{-2}\right).$$

Hence the number of $\mathcal{A}$-hard integers $n \le x$ is bounded by

$$\int_{n_0^-}^{x} \frac{x}{t}\, dS(t) \le S(x) + x\int_{n_0^-}^{x} \frac{S(t)}{t^2}\, dt \ll_a x^{1/(a-7/4)} + x\int_{n_0}^{x} t^{-2+1/(a-7/4)}\, dt \ll x n_0^{-1+1/(a-7/4)}$$
$$\ll x\exp\left(-\frac{a-11/4}{9(a-7/4)}(\log B)^3(\log\log B)^{-2}\right).$$

The number of passes through steps 2–5 is, again, $\ll \log n$. The oracle Dec Φ only needs to be queried once, for the original value of n, to supply the algorithm input. For divisors $n' \mid n$ it suffices to have a factorization of a multiple of $\phi(n')$ to perform steps 2–3, and we do have $\phi(n') \mid \phi(n)$. Step 2 and 3 each take $\ll B\log n\log\log n$ multiplications mod n, which can be seen as follows. Let $r_1 \cdot \ldots \cdot r_q$ be the original prime decomposition of $\phi(n)$. Then $\operatorname{ord}_n b$ computed in Step 2 is the smallest divisor d of $r_1 \cdot \ldots \cdot r_q$ such that b^d is 1 mod n, while Step 3 requires the computation of b^{d/r_i} for each $r_i \mid d$. We first show an upper bound $T(r_1, \ldots, r_q)$, for the number of multiplications mod n necessary to complete Step 2. For $q = 1$ we have $T(r_1) \ll \log r_1$. For $q = 2q'$ we can first consider all d of the form

$$d = r_1 \ldots r_{q'} d_2, \qquad d_2 \mid r_{q'+1} \ldots r_{2q'},$$

involving $\log(r_1 \ldots r_{q'}) + T(r_{q'+1}, \ldots, r_{2q'})$ multiplications, then, having found the minimal d_2, consider

$$d = d_1 d_2, \qquad d_1 \mid r_1 \ldots r_{q'},$$

involving $\log(r_{q'+1}, \ldots, r_{2q'}) + T(r_1, \ldots, r_{q'})$ multiplications, and then take $d = d_1 d_2$. Thus

$$T(r_1, \ldots, r_{2q'}) \le \log(r_1 \ldots r_{2q'}) + T(r_1, \ldots, r_{q'}) + T(r_{q'+1}, \ldots, r_{2q'}).$$

Hence $T(r_1, \ldots, r_q) \ll \log(r_1 \ldots r_q)\log q$. Similarly, to compute all the powers b^{d/r_i} it suffices to compute $b_1 = b^{d_1}$, $b_2 = b^{d_2}$, and then

$$b_1^{d_2/r_i} \qquad \text{for all} \qquad r_i \mid d_2$$

and

$$b_2^{d_1/r_i} \qquad \text{for all} \qquad r_i \mid d_1.$$

We then obtain an upper bound for the complexity of Step 3 in the same way as for Step 2. Step 4 takes w multiplications mod n. The complexity of Step 5 is $\ll (\log n)^{5+\varepsilon}$. Hence the total complexity of $\mathcal{A}$ for $n \le x$ is

$$\ll (\log x)^a(\log n)^{2+\varepsilon} + (\log x)^{a+2}(\log n)^{1+\varepsilon} + (\log n)^{6+\varepsilon}$$
$$\ll (\log x)^{a+3+\varepsilon}.$$

It suffices to set $a = M + 7/4$ and the proof of Theorem 4.1 is complete. □

5 Factorization by iterated use of the Φ oracle

Algorithm $\mathcal{A}_3$ has a much better estimate for hard numbers than $\mathcal{A}_1$ and $\mathcal{A}_2$, but it requires the Dec Φ oracle. Źrałek [18] was able to use iterated queries to the Φ oracle in place of the Dec Φ oracle. In our case, however, exceptional hard integers may appear, and it would be difficult to tell for how many n the value $\varphi(\dots\varphi(n)\dots)$ would belong to the exceptional set. (Problems related to pre-images of the totient function tend to be hard, cf., e.g., [7]). However, we propose a hybrid approach (algorithm $\mathcal{A}_4$) that employs both $\mathcal{A}_1$ and $\mathcal{A}_3$, and requires $\ll \log n$ queries to the oracle Φ. For this algorithm we obtain an estimate of the number of *-hard numbers as good as for $\mathcal{A}_1$, and for the number of hard numbers—as good as for $\mathcal{A}_3$. Let $\varphi_0(n) = n$ and let $\varphi_k(n) = \varphi(\dots\varphi(n)\dots)$ denote the k-th iteration of φ. In contrast to the previous algorithms, for this one we do not use $\mathcal{A}_0$ as an "outer shell", but inside $\mathcal{A}_4$.

Algorithm 4 ($\mathcal{A}_4$) Factorization based on the multiple use of the Φ oracle

Input Positive integer n, auxiliary parameters B, y, where $B \le y$.
Output Factorization of n.

1. Use the algorithm $\mathcal{A}_0(\mathcal{A}_1(B, y))$ with the Φ-oracle to obtain a complete or partial factorization of n.
2. If the obtained factorization contains a non-prime factor
$$n' > \exp\left(\frac{1}{9}(\log B)^3(\log\log B)^{-2}\right),$$
declare n hard and halt.
3. For all remaining non-prime factors n' compute $\varphi(n')$, $\varphi_2(n')$, …, $\varphi_k(n')$, where $\varphi_k(n') \le 2$. For $j = k-1, k-2, \dots, 0$ use the algorithm $\mathcal{A}_0(\mathcal{A}_3(B, y))$ and the known factorization of $\varphi_{j+1}(n')$ to factorize $\varphi_j(n')$.

Theorem 5.1. *We have, for arbitrary fixed $M \ge 4$, $\mathcal{A} = (\mathcal{A}_4, B, y)$, and appropriate choices of B and y:*
$$F(x, \mathcal{A}, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x\exp\left(-\frac{M^3(\log\log x)^3}{9(\log(M+2)+\log\log\log x)^2}\right)\right)$$
and
$$F^*(x, \mathcal{A}, \Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge x - O_M\left(x^{1.34/M}\right),$$
where $t_{\mathcal{O}} \ll \log x$ and $t_{\mathcal{A}} = O((\log x)^{M+5})$.

Proof of Theorem 5.1. The theorem essentially follows from Theorems 3.1 and 4.1, however we need to refer to their proofs when checking one assertion: the estimate for the number of $\mathcal{A}$-hard integers. Let x be large, $B = (\log x)^a$, $a = M - 7/4$, $y = B$, and let ε be small, given a. Let S denote the set of $\mathcal{A}^*$-hard integers. Again, the smallest element n_0 of S satisfies the bound
$$n_0 > \exp\left(\frac{1}{9}(\log B)^3(\log\log B)^{-2}\right).$$
Hence the number of $\mathcal{A}$-hard integers $n \le x$ is bounded by
$$\int_{n_0^-}^{x} \frac{x}{t}\,dS(t) \le S(x) + x\int_{n_0^-}^{x} \frac{S(t)}{t^2}\,dt \ll_a x^{4/(3a-9/4)+\varepsilon} + x\int_{n_0}^{x} t^{-2+4/(3a-9/4)}\,dt \ll x n_0^{-1+4/(3a-9/4)}$$
$$\ll x\exp\left(-\frac{3a-25/4}{9(3a-9/4)}(\log B)^3(\log\log B)^{-2}\right) \ll x\exp\left(-\frac{(M-1/3)(M+7/4)^3}{9(M+1)}(\log\log x)^3(\log\log B)^{-2}\right).$$
□

6 Oracles related to multiples of $\phi(n)$

Three of the algorithms presented above can also be used with slightly weaker oracles. Algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ can be run for n given any multiple D of $\phi(n)$, in which case the computational complexity grows by a factor of $\log D/\log x$. Likewise algorithm $\mathcal{A}_3$ works given a prime factorization of a multiple D of $\phi(n)$, whence the computational complexity grows by a factor of $(\log D/\log x)^{1+\varepsilon}$. Therefore D can be larger than an arbitrarily large fixed power of x. In fact it can be as large as $\exp((\log x)^{O(1)})$ for the algorithms to finish in polynomial time. However, algorithms $\mathcal{A}_0$ and $\mathcal{A}_4$ do depend on having the exact value of $\phi(n)$. We can therefore state variants of Theorems 3.1, 3.5 and 4.1 referring to "raw" algorithms $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ (without $\mathcal{A}_0$) and using weaker oracles, at the cost of restricting them to square-free integers.

Consider oracles $\operatorname{Mul}\Phi = \operatorname{Mul}^{M'}\Phi$ and $\operatorname{Dec}\operatorname{Mul}\Phi = \operatorname{Dec}\operatorname{Mul}^{M'}\Phi$ for some fixed positive M'. Let $\mathcal{S}(x)$ denote the number of square-free positive integers $n \le x$.

Theorem 6.1. *For arbitrary fixed $M \ge 4$, and appropriate choices of B, y and z, we have:*

$$F(x, \mathcal{A}_1, \operatorname{Mul}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge \mathcal{S}(x) - O_M\left(x(\log x)^{-6.5M}\right),$$

$$F^{\star}(x, \mathcal{A}_1, \operatorname{Mul}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge \mathcal{S}(x) - O_M\left(x^{1.34/M}\right),$$

and likewise

$$F(x, \mathcal{A}_2, \operatorname{Mul}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge \mathcal{S}(x)\left(1 - O_M((\log x)^{-6.5M})\right),$$

$$F^{\star}(x, \mathcal{A}_2, \operatorname{Mul}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge \mathcal{S}(x) - O_M\left(x^{1.34/M}\right),$$

and

$$F(x, \mathcal{A}_3, \operatorname{Dec}\operatorname{Mul}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge \mathcal{S}(x) - O_M\left(x\exp\left(-\frac{M^3(\log\log x)^3}{9(\log(M+2)+\log\log\log x)^2}\right)\right),$$

$$F^{\star}(x, \mathcal{A}_3, \operatorname{Dec}\operatorname{Mul}\Phi, t_{\mathcal{A}}, t_{\mathcal{O}}) \ge \mathcal{S}(x) - O_M\left(x^{1/M}\right),$$

where $t_{\mathcal{O}} = 1$, and $t_{\mathcal{A}} = O\left((\log x)^{M+M'+5}\right)$.

This essentially follows from the proofs of Theorems 3.1, 3.5 and 4.1 and the well known fact that $\mathcal{S}(x)$ is of the same order as x.

In the special case when n is a product of two primes, $n = pq$, the problem of factoring n with the oracle $\operatorname{Dec}\operatorname{Mul}\Phi$ is always solvable using algorithm $\mathcal{A}_3$, by (2). Using the oracle Φ this problem is trivial and the solution is well known. However, it is not trivial with the weaker oracle $\operatorname{Mul}\Phi$. We have

Theorem 6.2. *All except $O_M\left(x^{1/M}\right)$ integers of the form $n = pq \le x$ can be factored using algorithm $\mathcal{A}_1$ in time $t_{\mathcal{A}} = O\left((\log x)^{M+M'+5}\right)$ with one query to the oracle* $\operatorname{Mul}\Phi$.

Again, this essentially follows from the proof of Theorem 3.1, except we do not need to consider case (*ii*) of Lemma 3.2, and thus obtain a better exponent in the estimate of the number of hard integers.

Acknowledgement: The authors gratefully acknowledge the helpful remarks of the referees. In particular, the section on using weaker oracles was added to the paper as a result of one of the referees' suggestions.

References

[1] L. M. Adleman, K. S. McCurley, *Open problems in number theoretic complexity II.* In: Algorithmic Number Theory, First International Symposium, ANTS-I, Ithaca, NY, USA, 291–322 (1994), Springer-Verlag.

[2] N. C. Ankeny, *The least quadratic non residue*, Ann. of Math. 55 (1952), 65–72.

[3] E. Bach, *Discrete logarithms and factoring*, Technical report UCB/CSD-84-186, EECS Department, University of California, Berkeley. http://www.eecs.berkeley.edu/Pubs/TechRpts/1984/5973.html.
[4] S. Baier, *On the least n with $\chi(n) \neq 1$*, Quart. J. Math. 57 (2006), 279–283.
[5] R. Crandall, C. Pomerance, *Prime Numbers — A Computational Perspective*, Second Edition, Springer 2005.
[6] K. Durnoga, J. Pomykała, *Large sieve, Miller-Rabin compositeness witnesses and integer factoring problem*, Fundamenta Informaticae, 156(2), 2017, 179–185.
[7] K. Ford, *The number of solutions of $\phi(x) = m$*, Ann. of Math. (2) 150 (1999), no. 1, 283–311.
[8] S. Konyagin, C. Pomerance, *On primes recognizable in deterministic polynomial time*, in: The mathematics of Paul Erdos, R. L. Graham and J. Nesetril (eds.), Springer-Verlag, 1997, 176–198.
[9] S. Landau, *Some remarks on computing the square parts of integers*, Inf. Comput. 78(3), 1988, 246–253.
[10] Y. K. Lau, J. Wu, *On the least quadratic non-residue*, Int. J. Number Theory 04, 423 (2008).
[11] H. W. Lenstra, Jr., C. Pomerance, *Primality testing with Gaussian periods*, in: Manuscript, math.dartmouth.edu/ carlp, 2005. - See more at: http://www.ams.org/journals/mcom/2015-84-291/S0025-5718-2014-02840-8/#sthash.kmasMqnp.dpuf
[12] G. L. Miller, *Riemann's Hypothesis and tests for primality*, Journal of Computer and System Sciences 13(3), (1976), 300–317.
[13] F. Morain, G. Renault, B. Smith, *Deterministic factoring with oracles* https://hal.inria.fr/hal-01715832/document
[14] J. Pomykała, B. Źrałek, *On reducing factorization to the discrete logarithm problem modulo a composite*, Comput. Complex 21 (2012), 421–429.
[15] M. O. Rabin, *Probabilistic algorithm for testing primality*, Journal of Number Theory 12(1), 1980, 128–138.
[16] A. Schönhage, V. Strassen, *Schnelle Multiplikation großer Zahlen*, Computing 7 (1971), 281–292.
[17] G. Tenenbaum, *Introduction to Analytic and Probabilistic Number Theory: Third Edition*, Graduate Studies in Mathematics 163, AMS, 2015.
[18] B. Źrałek, *A deterministic version of Pollard's $p - 1$ algorithm*, Math. of Comp. 79 (2010), 513–533.