Journal of Complexity 36 (2016) 1–30

Contents lists available at ScienceDirect

Journal of Complexity

journal homepage: www.elsevier.com/locate/jco

Even faster integer multiplication

David Harvey [a], Joris van der Hoeven [b,*], Grégoire Lecerf [b]

[a] School of Mathematics and Statistics, University of New South Wales, Sydney NSW 2052, Australia
[b] CNRS, Laboratoire d'informatique, École polytechnique, 91128 Palaiseau Cedex, France

ARTICLE INFO

Article history:
Received 29 June 2015
Accepted 18 February 2016
Available online 15 March 2016

Keywords:
Integer multiplication
Algorithm
Complexity bound
FFT

ABSTRACT

We give a new algorithm for the multiplication of n-bit integers in the bit complexity model, which is asymptotically faster than all previously known algorithms. More precisely, we prove that two n-bit integers can be multiplied in time $O(n \log n\, K^{\log^* n})$, where $K = 8$ and $\log^* n = \min\left\{k \in \mathbb{N} : \log^{\circ k} n \leqslant 1\right\}$. Assuming standard conjectures about the distribution of Mersenne primes, we give yet another algorithm that achieves $K = 4$. The fastest previously known algorithm was due to Fürer, who proved the existence of a complexity bound of the above form for some finite K. We show that an optimised variant of Fürer's algorithm achieves only $K = 16$, suggesting that our new algorithm is faster than Fürer's by a factor of $2^{\log^* n}$.

1. Introduction

Let $\mathsf{I}(n)$ denote the cost of multiplying two n-bit integers in the deterministic multitape Turing model [37] (commonly called "bit complexity"). Previously, the best known asymptotic bound for $\mathsf{I}(n)$ was due to Fürer [18,19]. He proved that there is a constant $K > 1$ such that

$$\mathsf{I}(n) = O(n \log n\, K^{\log^* n}), \tag{1}$$

* Corresponding author.
E-mail addresses: d.harvey@unsw.edu.au (D. Harvey), vdhoeven@lix.polytechnique.fr (J. van der Hoeven), lecerf@lix.polytechnique.fr (G. Lecerf).

http://dx.doi.org/10.1016/j.jco.2016.03.001
0885-064X/

where $\log x = \ln x$ denotes the natural logarithm of x and $\log^* x$ the iterated logarithm, i.e.,

$$\log^* x := \min\{k \in \mathbb{N} : \log^{\circ k} x \leqslant 1\}, \tag{2}$$
$$\log^{\circ k} := \log \circ \underset{k\times}{\cdots} \circ \log,$$

for any $x \in \mathbb{R}$ with $x > 0$. The main contribution of this paper is a new algorithm that yields the following improvement.

Theorem 1. *For $n \to \infty$ we have*

$$\mathsf{I}(n) = O(n \log n\, 8^{\log^* n}).$$

Fürer suggested several methods to minimise the value of K in his algorithm, but did not give an explicit bound for K. In Section 7 of this paper, we outline an optimised variant of Fürer's algorithm that achieves $K = 16$. We do not know how to obtain $K < 16$ using Fürer's approach. This suggests that the new algorithm is faster than Fürer's by a factor of $2^{\log^* n}$.

The idea of the new algorithm is remarkably simple. Given two n-bit integers, we split them into chunks of exponentially smaller size, say around $\log n$ bits, and thus reduce to the problem of multiplying integer polynomials of degree $O(n/\log n)$ with coefficients of bit size $O(\log n)$. We multiply the polynomials using discrete Fourier transforms (DFTs) over $\mathbb{C}$, with a working precision of $O(\log n)$ bits. To compute the DFTs, we decompose them into "short transforms" of exponentially smaller length, say length around $\log n$, using the Cooley–Tukey method. We then use Bluestein's chirp transform to convert each short transform into a polynomial multiplication problem over $\mathbb{C}$, and finally convert back to integer multiplication via Kronecker substitution. These much smaller integer multiplications are handled recursively.

The algorithm just sketched leads immediately to a bound of the form (1). A detailed proof is given in Section 4. We emphasise that the new method works directly over $\mathbb{C}$, and does not need special coefficient rings with "fast" roots of unity, of the type constructed by Fürer. Optimising parameters and keeping careful track of constants leads to Theorem 1, which is proved in Section 6. We also prove the following conditional result in Section 9, where we recall that a Mersenne prime is a prime of the form $p = 2^q - 1$.

Theorem 2. *Let $\pi_m(x)$ denote the number of Mersenne primes less than x. If the function $x \mapsto \pi_m(x)/\log\log x$ is bounded both from above and from below on $(3, \infty)$, then*

$$\mathsf{I}(n) = O(n \log n\, 4^{\log^* n}).$$

The assumption on $\pi_m(x)$ is a weakening of the Lenstra–Pomerance–Wagstaff conjecture on the distribution of Mersenne primes. The idea of the algorithm is to replace the coefficient ring $\mathbb{C}$ by the finite field $\mathbb{F}_p[\mathrm{i}]$; we are then able to exploit fast algorithms for multiplication modulo numbers of the form $2^q - 1$.

An important feature of the new algorithms is that the same techniques are applicable in other contexts, such as polynomial multiplication over finite fields. Previously, no Fürer-type complexity bounds were known for the latter problem. The details are presented in the companion paper [23].

In the remainder of this section, we present a brief history of complexity bounds for integer multiplication, and we give an overview of the paper and of our contribution. More historical details can be found in books such as [54, Chapter 8].

1.1. Brief history and related work

Multiplication algorithms of complexity $O(n^2)$ in the number of digits n were already known in ancient civilisations. The Egyptians used an algorithm based on repeated doublings and additions. The Babylonians invented the positional numbering system, while performing their computations