Version: 1.0, February 1999

Datalogi 2P: Introduction to Algorithmics
Exercise O1, Spring 1999

Jyrki Katajainen

Department of Computing, University of Copenhagen
DK-2100 Copenhagen East, Denmark
E-mail: `jyrki@diku.dk`

This exercise consists of two parts: A and B. You should solve A **or** B to get your answer accepted. (If you are desperate, you can of course solve both parts.) You should return your answer to our study administration by **Wednesday 17 February 1999** at 12.00.

Assume that the word size of your computer is λ. Further, assume that the primitives for double precision arithmetic are provided, namely, functions for double-precision addition, subtraction, and multiplication. For example, for two words x and y their product $x \cdot y$ is a pair of words $\langle a, b \rangle$, where a is the higher-order word of the product and b the lower-order word.

Part A

Consider the problem of multiplying two (positive) multi-word integers.

a) Write pseudocode for the standard school multiplication algorithm and analyse the number of double-precision arithmetic operations required by the algorithm in the worst case as exactly as possible. Observe that a description of the school algorithm appeared already in al-Khwārizmī's texts (for further details, see [2]).

b) Give an efficient divide-and-conquer algorithm for multiplying two multi-word integers and analyse its worst-case performance, again as exactly as possible. As a hint look at Problem 32-1 in [3]. Apparently the first divide-and-conquer multiplication algorithm was devised by Karatsuba and Ofman [4].

c) Based on your analysis, what is the *crossover point* of the two algorithms, that is, when is the divide-and-conquer algorithm faster than the school algorithm.

d) Consider a hybrid method that solves the large subproblems by using the divide-and-conquer method and small subproblems by the school method. Determine analytically the optimal *cutoff point*, when one should switch to the school method.

e) Implement both the school and divide-and-conquer multiplication algorithms and compare their performance experimentally. To make your implementation task easier, a driver program (in C++), a make file, a shell script, and a gnuplot script are provided via the Datalogi-2P home page at

`http://www.diku.dk/teaching/1del/dat2p-algoritmik/`

Part B

In [8] (see also [1, 5, 6, 7]) Schönhage and Strassen presented an asymptotically fast multiplication algorithm. Sketch the main idea of this algorithm and analyse its asymptotic behaviour when multiplying two multi-word integers. What can you say about the constant factors in the function describing the performance of the algorithm?

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley Publishing Company, Reading (1974).

[2] A. Allard, *Muhammad ibn Mūsā Al-Khwārizmī: Le Calcul Indien*, Société des Études Classiques, Namur (1992).

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, The MIT Press, Cambridge (1990).

[4] А. Карацуба and Ю. Офман, Умножение многозначных чисел на автоматах, *Доклады Академии наук СССР* **145** (1962), 293–294. English translation: A. Karatsuba and Yu. Ofman, Multiplication of multidigit numbers on automata, *Soviet Physics—Doklady* **7** (1963), 595–596.

[5] D. E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, 3. edition, Addison-Wesley Publishing Company, Reading (1998).

[6] A. Schönhage, Asymptotically fast algorithms for numerical multiplication and division of polynomials with complex coefficients, in *Proceedings of the European Computer Algebra Conference*, Lecture Notes in Computer Science **144**, Springer-Verlag, Berlin/Heidelberg (1982), 3–15.

[7] A. Schönhage, A. F. W. Grotefeld, and E. Vetter, *Fast Algorithms: A Multitape Turing Machine Implementation*, B•I•Wisserschaftsverlag, Mannheim (1994).

[8] A. Schönhage and V. Strassen, Schenelle Multiplikation großen Zahlen, *Computing* **7** (1971), 281–292.