José Carlos Santos
Universidade
do Porto
jcsantos@fc.up.pt

COMO DIVIDIR E CONQUISTAR A MULTIPLICAÇÃO

Podia pensar-se que a maneira como todos aprendemos a multiplicar números naturais é a mais rápida de todas. Mas não é.

A maneira como aprendemos a multiplicar números naturais no primeiro ciclo do Ensino Básico é usada há séculos em muitas regiões da Terra. É tão comum que possivelmente muita gente pensará que é a maneira de multiplicar quaisquer dois números naturais, embora, de facto, haja mais algoritmos tão ou mais antigos, como o método da gelosia,[1] que vem dos árabes e que foi introduzido na Europa por Fibonacci, ou o método dos camponeses,[2] que não exige conhecer a tabuada.

Ao multiplicarmos dois números de n algarismos cada pelo método tradicional, temos de levar a cabo a multiplicação de cada algarismo de cada número por cada algarismo do outro número. Isso leva a um total de n^2 multiplicações. Será necessário ainda fazer mais algumas adições para completar o cálculo.

Uma questão natural a colocar aqui é a de saber se é ou não o método mais rápido de levar a cabo este cálculo. Visto que é um algoritmo muito antigo, é razoável pensar que, caso houvesse um algoritmo mais rápido, no sentido de o número de operações a levar a cabo ser de uma ordem de grandeza menor do que no caso do algoritmo tradicional, este já teria sido descoberto há muito. Aqui, "ordem de grandeza menor" significa um algoritmo cujo número de passos seja proporcional, não a n^2 operações, como no caso do algoritmo tradicional, mas, por exemplo, proporcional a $n^{1,8}$ operações.

A título de exemplo, consideremos o algoritmo de Euclides para calcular o máximo divisor comum de dois números naturais. Surge nos *Elementos* de Euclides, que foram escritos por volta de 300 a.C. E continua, ao fim destes milénios a ser o algoritmo mais rápido para o fim a que se destina. De facto, em 1967 o físico Josef Stein publicou um artigo (veja-se [3]) onde divulgava outro algoritmo para o cálculo do máximo divisor comum de dois números que é mais rápido do que o de Euclides, mas somente no sentido de exigir cerca de 60% dos cálculos, não no sentido de o número de cálculos ser de uma ordem de grandeza menor do que no caso do algoritmo de Euclides.[3]

Na década de 1950, o matemático russo Andrey Kolmogorov conjeturou que o algoritmo usual de multiplicação é o melhor que pode existir, no sentido atrás descrito. Em 1960, organizou um seminário de Cibernética

[1] Veja-se *O Método da Gelosia para Multiplicações*, de Kleber Kilhian: *https://www.obaricentrodamente.com/2011/12/o-metodo-da-gelosia-para-multiplicacoes.html*

[2] Veja-se *Método da Multiplicação dos Camponeses Russos*} de Kleber Kilhian: *https://www.obaricentrodamente.com/2011/03/metodo-da-multiplicacao-dos-camponeses.html*

[3] Veja-se *Binary GCD Algorithm*: *https://iq.opengenus.org/binary-gcd-algorithm/*

A. Karatsuba

na Faculdade de Mecânica e Matemática da Universidade de Moscovo, que tinha como objetivo demonstrar essa conjetura. Para sua grande surpresa (imagina-se), o que aconteceu foi que Anatoly Karatsuba, na altura com 23 anos, demonstrou o contrário, ao encontrar um algoritmo mais eficiente para a multiplicação. Karatsuba expôs o seu algoritmo a Kolmogorov. Na sessão seguinte do seminário, este divulgou o algoritmo e, em seguida, declarou o seminário terminado (veja-se [1, §6]). Karatsuba viria a ser um especialista em Teoria Analítica dos Números.

Dois anos mais tarde, foi publicado o artigo [2], onde era exposto o algoritmo de Karatsuba. De facto, Karatsuba não esteve envolvido na publicação do artigo. Foi escrito por Kolmogorov (provavelmente com a colaboração de Yuri Ofman) e Karatsuba só tomou conhecimento da publicação do artigo ao receber as separatas que lhe eram devidas, na sua qualidade de (suposto) autor.

Vejamos então como funciona este algoritmo. Queremos multiplicar dois números naturais a e b, com $a \geqslant b$. Vamos supor que os escrevemos em base 10 (o algoritmo funciona em qualquer base) e que a se escreve em base 10 com n algarismos. Toma-se um número natural $m < n$ e escreve-se

$$a = 10^m a_1 + a_0 \quad \text{e} \quad b = 10^m b_1 + b_0,$$

com $a_1, a_0, b_1, b_0 \in \mathbb{N}$. Então

$$ab = 10^{2m} a_1 b_1 + 10^m (a_1 b_0 + a_0 b_1) + a_0 b_0.$$

Isto exige então quatro multiplicações. A ideia de Karatsuba consistiu em aplicar a seguinte relação:

$$a_1 b_0 + a_0 b_1 = (a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1. \qquad (1)$$

Parece que estamos aqui a substituir duas multiplicações por três, mas é preciso ter em conta que as multiplicações $a_0 b_0$ e $a_1 b_1$ já tinham sido feitas anteriormente. Assim sendo, o cálculo de ab envolve somente três multiplicações: $a_0 b_0$, $a_1 b_1$ e $(a_0 + a_1)(b_0 + b_1)$. Observe-se que (1) também envolve subtrações, mas há aqui uma ideia implícita (e intuitiva) segundo a qual somas e subtrações têm um peso insignificante relativamente à multiplicação.

Vejamos, por exemplo, como multiplicar 13 579 por 2 468. Será usado $m = 3$. Temos então $13\,579 = 13 \times 10^3 + 579$ e $2\,468 = 2 \times 10^3 + 468$. Então, seguindo o algoritmo, calculam-se

- $13 \times 2 = 26$
- $579 \times 468 = 270\,972$
- $(13 + 579)(2 + 468) - 26 - 270\,972 = 7\,242.$

Então

$$\begin{aligned} 13\,579 \times 2\,468 &= 26 \times 10^6 + 7\,242 \times 10^3 + 270\,972 \\ &= 26\,000\,000 + 7\,242\,000 + 270\,972 \\ &= 33\,512\,972. \end{aligned}$$

Uma análise cuidadosa ao algoritmo de Karatsuba, revela que a sua aplicação exige cerca de $n^{\log_2 3}$ multiplicações de números com um único algarismo. Como $\log_2 3 \approx 1{,}58 < 2$, este algoritmo é claramente melhor do que o algoritmo tradicional.

Em informática teórica, o algoritmo de Karatsuba faz parte de uma família de algoritmos que se designa por "Divisão e conquista". São algoritmos que, para resolver um problema, o separam em dois ou mais subproblemas, resolvem estes problemas e empregam as soluções para resolver o problema original. Por vezes, aplica-se a designação "Divisão e conquista" a algoritmos que começam, não por reduzir o problema original a vários problemas mais simples, mas a um problema mais simples. É o caso do algoritmo de Euclides: para encontrar

o máximo divisor comum de dois números naturais m e n, com $m > n$, o primeiro passo consiste em encontrar o máximo divisor de n e do resto da divisão de m por n.

O algoritmo de Karatsuba não é a última palavra sobre este tópico. O que aconteceu foi que a descoberta deste algoritmo estimulou a procura de outros que fossem ainda mais rápidos. Logo em 1963 surgiu o algoritmo de Toom-Cook, que exige cerca de $n^{\log_3 5}$ multiplicações de números formados por um algarismo. Como $\log_3 5 \approx 1{,}46$, é mais rápido do que algoritmo de Karatsuba. E, em 1971, surgiu o algoritmo de Schönhage-Strassen, que exige $n \log(n) \log(\log(n))$ multiplicações; é menor do que $n^{\log_3 5}$ para n suficientemente grande. Já se passaram mais de 50 anos e ainda não surgiu nenhum algoritmo mais rápido.

E porque é que falha o argumento segundo o qual se houvesse um algoritmo melhor, então já teria sido descoberto há muito? Acontece que os algoritmos atrás mencionados não justificam o esforço para lidar com números à escala humana. De facto, levam a mais cálculos e não a menos se forem aplicados a números pequenos. É ao fazer-se a multiplicação de números com dezenas ou centenas de algarismos que a vantagem do uso daqueles algoritmos se torna avassaladora.

REFERÊNCIAS

[1] Anatoly A. Karatsuba, "The complexity of computations, *Proceedings of the Sketlov Institute of Mathematics*, 211, pp. 169-183, 1995

[2] Anatoly A. Karatsuba e Yuri Ofman (1962). "Multiplicação de números com muitos algarismos por computadores automáticos" (em russo), *Proceedings of the USSR Academy of Sciences*, 145: 293–294, 1962. Tradução para inglês em *Physics-Doklady*, 7, pp. 595-596, 1963

[3] Josef Stein, "Computational problems associated with Racah algebra", *Journal of Computational* **1** (3), pp. 397-405, 1967