Analisi d'un algorisme de multiplicació d'enters

Jordi Marco
José Luis Balcázar

Report LSI–93–11–T

Anàlisi d'un algorisme de multiplicació d'enters

Jordi Marco i José L. Balcázar

Resum:

Es proposa un algorisme de multiplicació d'enters basat en l'esquema "divideix i vençe", però diferent del de Karatsuba i Ofman. Es fa l'anàlisi de la seva complexitat. Es troba que, encara que sembli millor que els ja coneguts si el cost de la suma es considera constant, fent la hipótesi, més realista, que la suma requereix temps lineal, el cost total puja fins a quadràtic.

Presentació

En la recerca per tenir el menor cost en la multiplicació de enters grans, han estat descoberts, durant molt temps, molts algorismes.

Un d'aquests, i potser el més important i difós actualment, és l'algorisme descobert per A. Karatsuba i Y. Ofman [1]. Aquest algorisme pren el temps $O(n^{\log_2 3})$, sent n el nombre de bits dels enters a multiplicar, es a dir $O(n^{1.59})$ aproximadament.

Posteriorment i basant-se en aquest, es van proposar altres algorismes més eficaços. Es poden veure referències adients a [2].

L'algorisme de menor cost descobert fins ara és el descrit per A. Schönhage i V. Strassen [3], que es capaç de calcular la multiplicació de dos enters de n bits en un temps $O(n \log n \log\log n)$.

A continuació presentem un algorisme de multiplicació que, al igual que l'algorisme de Karatsuba i Ofman, es basa en la tècnica de "divideix i vençe", però difereix d'aquest en què només divideix un dels dos nombres i sols fa dues crides recursives, és a dir, sols dues noves multiplicacions. Com a conseqüència, es pot esperar un millor comportament, i així ho demostrarem sota la hipòtesi de el cost de la suma sigui constant.

Però, sota la hipòtesi, més realista, que la suma pren cost lineal, l'anàlisi mostra que aquest algorisme requereix un temps $O(n^2)$, i per tant no és millor que els algorismes elementals. La causa d'aquest comportament és doncs el cost de les operacions auxiliars, i no pas la pròpia recurrència.

L'algorisme

Siguin x i y dos enters de n bits a multiplicar, amb $x = 2^{n/2}a + b$ i $y = 2^{n/2}c + d$. El producte, segons el algorisme de Karatsuba i Ofman, serà:

$$xy = 2^n ac + 2^{n/2}((a-b)(d-c) + ac + bd) + bd$$

Com es pot observar, aquest algorisme descomposa la multiplicació en tres noves multiplicacions combinades amb addicions, substracions i desplaçaments de bits.

L'algorisme que presentem ara calcula la multiplicació de la següent manera:

$$xy = x(2^{n/2}c + d) = xc2^{n/2} + xd$$

sent igualment x i y dos enters de n bits a multiplicar amb el valor expresat anteriorment.

Observem que, per tal d'aconseguir el resultat de la multiplicació, sols transforma aquesta en dues noves multiplicacions envers de les tres que utilitza l'algorisme de Karatsuba i Ofman.

Anàlisi

A continuació s'estudia el cost real per tal de descobrir si efectivament el cost d'aquest nou algorisme és menor. Per tal d'efectuar aquest estudi partirem de dues premises diferents:

a. Que el cost de la suma és constant (irrellevant).
b. Que el cost de la suma és lineal.

Sigui $T(N,n)$ el temps necesari per a multiplicar dos enters de longitud N i n respectivament, on N és el tamany de l'enter que no es divideix i n el tamany de l'altre. L'anàlisi indicarà que és preferible que n sigui el major de tots dos. Sense pèrdua de generalitat, suposem que n és una potència de 2.

Veiem que $T(N,n)$ és una constant quan $n = 1$, ja que si l'enter de tamany n té com a valor 0 la multiplicació tindrà com a resultat 0 i, si te com a valor 1, el seu resultat serà el valor de l'altre enter.

Quan $n > 1$, s'han de calcular recursivament dues noves multiplicacions on els enters tindràn, com a molt, tamany N i $n/2$ respectivament. A més haurem d'afegir els costos adicionals de les sumes i els desplaçaments amb el que obtindrem les dues següents eqüacions de recurrència depenents de cada una de les dues premises.

Premisa a: El cost de la suma és irrellevant

$$T(N,n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ 2T(N,n/2) + n & \text{si } n > 1 \end{cases}$$

En aquest cas el cost de l'algorisme es redueix a $O(n \log n)$. Ometem la resolució de la recurrència, que es pot trobar, per exemple, en [4]. Amb aquesta premisa obtenim un cost inferior al presentat anteriorment, el què pot fer pensar que hem trobat un algorisme meravellós, senzill i ràpid, però ara veurem com s'incrementa el cost amb el fet que la suma tingui un cost lineal respecte a l'enter més gran.

Premisa b: El cost de la suma és lineal, $O(N)$

$$T(N,n) \leq \begin{cases} O(1) & \text{si } n = 1 \\ 2T(N,n/2) + n + N & \text{si } n > 1 \end{cases}$$

Expandint la recurrencia de $T(N,n)$ tants cops com sigui posible s'obté:

$$T(N,n) \leq 2^m \cdot T(N,n/2^m) + \sum_{i=0}^{m-1} 2^i \cdot (N + n/2^i)$$

que és equivalent a:

$$T(N,n) \leq 2^m \cdot T(N,n/2^m) + \sum_{i=0}^{m-1} 2^i \cdot n/2^i + \sum_{i=0}^{m-1} 2^i \cdot N$$

on, per $m = \log_2 n$, el primer terme és $O(n)$ doncs $T(N,n/2^m)$ és constant, i el segon suma $O(n \log n)$ pel mateix càlcul que a la premisa anterior. S'obté:

$$T(N,n) \leq O(n \log n) + \sum_{i=0}^{m-1} 2^i \cdot N$$

Desenvolupant el sumatori, que es la suma d'una progresió geomètrica, obtenim:

$$T(N,n) \leq O(n \log n) + N \cdot (2^m - 1)/(2 - 1)$$

és a dir, de nou per $m = \log_2 n$,

$$T(N,n) \leq O(n \log n) + N \cdot (2^{\log n} - 1) = O(n \log n) + O(N \cdot n)$$

Així doncs, amb enters de la mateixa mida, la n inicial es igual a N, obtenint com a cost final $O(N^2)$. El cost és en qualsevol cas el màxim entre $O(n \log n)$ i $O(N \cdot n)$, i és $O(n \log n)$ sols si N es prou petit.

Es pot pensar que hi hagi alguna mena d'equilibri, y que es pugui aconseguir un cost inferior si la recurrència no s'expandeix fins al final. Això correspon amb un algorisme en qué, quan l'enter que decreix és prou petit (però, no necessàriament constant), no es fan més crides recursives, fent servir l'algorisme quadràtic usual. De l'anàlisi es desprén que, aleshores, el cost és encara quadràtic, degut al primer terme de la recurrència. En efecte, si a

$$T(N,n) \leq 2^m \cdot T(N,n/2^m) + \sum_{i=0}^{m-1} 2^i \cdot (N + n/2^i)$$

el terme $T(N, n/2^m)$ no és constant, sinò que val $N \cdot n/2^m$), que multiplicat per 2^m és ja $N \cdot n$, independentment del valor de m.

Així doncs, aquest algorisme, que semblava prometre un cost inferior a l'algorisme de Karatsuba i Ofman, té un cost igual al de l'algorisme tradicional de la multiplicació que tots coneixem des de que érem petits.

Agraïment

A en Ricard Gavaldà, pels seus suggeriments i comentaris.

Referències

[1] A. Karatsuba, Y. Ofman: "Multiplication of multidigit numbers on automata", *Dokl. Akad. Nauk SSSR* 145 (1962), 293–294 (en rus); trad. ang.: *Sov. Phys. Dokl.* 7 (1963), 595–596.

[2] G. Brassard, S. Monet, D. Zuffellato: "Algorithmes pour l'arithmétique des trés grands entiers", *Technique et Science Informatiques* 5 (1986), 89–103.

[3] A. Schönhage, V. Strassen: "Schnelle Multiplikation grosser Zahlen", *Computing* 7 (1971), 281–292.

[4] A. Aho, J. Hopcroft, J. Ullman: *The Design and Analysis of Computer Algorithms*, Addison-Wesley (1974).