In the name of God

Electronic advanced algorithms-4012

Homework-1: Divide and Conquer

Professor: Dr. Farahani

TA: Sara Charmchi

Student: Mehdi Khaefi

Student-id: 401422072

1. Divide-and-conquer multiplication

There is a faster way to multiply, though, called the *divide-and-conquer* approach.

Algorithm

With divide-and-conquer multiplication, we split each of the numbers into two halves, each with $n/2$ digits. I'll call the two numbers we're trying to multiply a and b, with the two halves of a being a_L (the left or upper half) and a_R (the right or lower half) and the two halves of b being b_L and b_R.

Basically, we can multiply these two numbers as follows.

$$
\begin{aligned}
ab &= (a_L\, 10^{n/2} + a_R)\ (b_L\, 10^{n/2} + b_R) \\
&= a_L\, b_L\, 10^n + a_L\, b_R\, 10^{n/2} + a_R\, b_L\, 10^{n/2} + a_R\, b_R \\
&= a_L\, b_L\, 10^n + (a_L\, b_R + a_R\, b_L)\ 10^{n/2} + a_R\, b_R
\end{aligned}
$$

We can reduce the number of $n/2$-digit multiplications from four to *three*!

The idea works as follows: We're trying to compute

$$a_L\, b_L\, 10^n + (a_L\, b_R + a_R\, b_L)\ 10^{n/2} + a_R\, b_R$$

What we'll do is compute the following three products using recursive calls.

$$
\begin{aligned}
x_1 &= a_L\, b_L \\
x_2 &= a_R\, b_R \\
x_3 &= (a_L + a_R)\ (b_L + b_R)
\end{aligned}
$$

These have all the information that we want, since the following is true.

$$
\begin{aligned}
& x_1\, 10^n + (x_3 - x_1 - x_2)\ 10^{n/2} + x_2 \\
&= a_L\, b_L\, 10^n + ((a_L\, b_L + a_L\, b_R + a_R\, b_L + a_R\, b_R) - a_L\, b_L - a_R\, b_R)\ 10^{n/2} + a_R\, b_R \\
&= a_L\, b_L\, 10^n + (a_L\, b_R + a_R\, b_L)\ 10^{n/2} + a_R\, b_R
\end{aligned}
$$

And we already reason that this last is equal to the product of a and b.

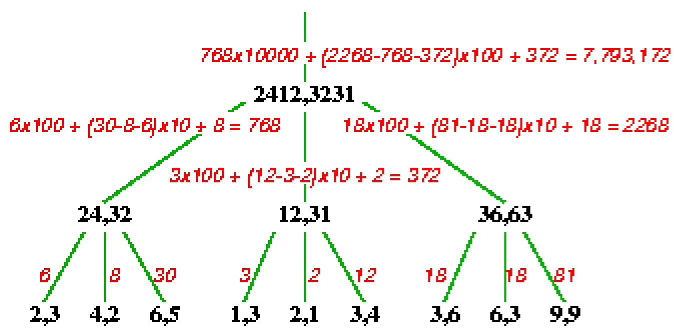
Pseudocode: Divide Conquer Combine

```
BigInteger multiply(BigInteger a, BigInteger b) {
    int n = max(number of digits in a, number of digits in b)
    if(n == 1) {
        return a.intValue() * b.intValue();
    } else {
        BigInteger aR = bottom n/2 digits of a;
        BigInteger aL = top remaining digits of a;
        BigInteger bR = bottom n/2 digits of b;
        BigInteger bL = top remaining digits of b;
        BigInteger x1 = Multiply(aL, bL);
        BigInteger x2 = Multiply(aR, bR);
        BigInteger x3 = Multiply(aL + aR, bL + bR);
        return x1 * pow(10, n) + (x3 - x1 - x2) * pow(10, n / 2) + x2;
    }
}
```

2. The smallest sub-problem:

The smallest sub-problem in this method is multiplying two one-digit numbers.

Let's do an actual multiplication to illustrate how this works. I'm going to draw a recursion tree, labeling the edges with the final values computed by each node of the tree.



4. Time complexity analysis

For n digit integer, we have to perform 3 multiplications of integers of size (n / 2). Recurrence equation for this problem is given as,

$T(n) = 3T(n/2)$, if $n > 1$

$T(n) = 1$, if $n = 1$

Proof:

$T(n) = 3T(n/2)$ **… (3)**

Let us solve this recurrence by an iterative approach. Substitute n by n/2 in Equation (3)

$T(n/2) = 3T(n/4)$ **… (4)**

Put this value in Equation (3),

$T(n) = 3(3T(n/4)) = 3^2T(n/2^2)$ **… (5)**

Substitute n by n/2 in Equation (4)

$T(n/4) = 3T(n/8)$

Put this value in Equation (3)

$T(n) = 3(3^2T(n/8)) = 3^3T(n/2^3)$ **… (6)**

.

.

.

.

After k iterations,

$T(n) = 3^kT(n/2^k)$ **…(7)**

Every time number of digits in number reduces by factor 2, so it can go as deep as $\log_2 n$, So, $k = \log_2 n \Rightarrow n = 2^k$

Thus from equation (5), $T(n) = 3^k T(2^k / 2^k)$

$T(n) = n^{\log_2 3} \times T(1)$ ($\because n^{\log_b a} = a^{\log_b n}$)

T (1) = 1(Only one multiplication is required to multiply two numbers of digits 1)

So, $T(n) = n^{\log_2 3} = O(n^{1.58})$

Grade school method multiplies each digit of the multiplier with each digit of the multiplicand. So for each digit of the multiplier, n multiplications are performed with multiplicand.

This is done each of the n bits of the multiplier. So running time of that method was $O(n^2)$, whereas the divide and conquer approach reduces the running time to $O(n^{1.58})$. For large numbers, the difference becomes significant.

References:

http://www.cburch.com/csbsju/cs/160/notes/31/1.html

https://codecrucks.com/large-integer-multiplication-using-divide-and-conquer/