Algorithm and Data Structure Analysis (ADSA)

Lecture 4: Karatsuba Multiplication

(Book Chapter 1)

Questions

- What is a recursive algorithm?
- State the basic feature for recursive school multiplication algorithm.
- Give the recursive formula for this algorithm.
- How did we get an upper bound for the runtime of this algorithm?

Overview

- Addition (last Thursday)
- School Method of Multiplication (last Thursday)
- Recursive Version of the School Method (last Tuesday)
- Karatsuba Multiplication

Karatsuba Multiplication

- Goal: Construct a faster recursive multiplication algorithm

$$a = a_1 \cdot B^k + a_0$$

$$b = b_1 \cdot B^k + b_0$$

$$a \cdot b = a_1 \cdot b_1 \cdot B^{2k} + (a_1 \cdot b_0 + a_0 \cdot b_1) \cdot B^k + a_0 \cdot b_0$$

$$\begin{aligned} = \quad & a_1 \cdot b_1 \cdot B^{2k} \\ & +((a_1 + a_0) \cdot (b_1 + b_0) - (a_1 \cdot b_1 + a_0 \cdot b_0)) \cdot B^k \\ & +a_0 \cdot b_0 \end{aligned}$$

Seems to be more complicated.

What do we gain from that?

Karatsuba Multiplication

$$
\begin{aligned}
p \quad = \quad & a_1 \cdot b_1 \cdot B^{2k} \\
& +((a_1 + a_0) \cdot (b_1 + b_0) - (a_1 \cdot b_1 + a_0 \cdot b_0)) \cdot B^k \\
& +a_0 \cdot b_0
\end{aligned}
$$

Observations:

- Compare to recursive school method.
- 3 multiplications (instead of 4).
- 6 additions (instead of 3).
- Multiplications are more costly than additions (previously n^2 versus n)

Karatsuba Algorithm

Algorithm (Karatsuba multiplication):

Recursive calls

1. Split a and b to obtain a_1, a_0, b_1, and b_0.
2. Compute the three products

$$p_2 = a_1 \cdot b_1, \quad p_0 = a_0 \cdot b_0, \quad p_1 = (a_1 + a_0) \cdot (b_1 + b_0)$$

3. Add the aligned products to obtain

$$p = a \cdot b = p_2 \cdot B^{2k} + (p_1 - (p_2 + p_0)) \cdot B^k + p_0$$

For n ≤ 3: use school method

For n ≥ 4: use these three steps

Theorem:

Let $T_K(n)$ be the maximal number of primitive operations by the Karatsuba algorithm. Then

$$T_K(n) \le \begin{cases} 3n^2 + 2n, & \text{if } n \le 3 \\ 3 \cdot T_K(\lceil n/2 \rceil + 1) + 6 \cdot 2 \cdot n & \text{if } n \ge 4 \end{cases}$$

Proof:

- $n \le 3$: School method.
- Splitting up the numbers does not requires primitive operations.
- Each subproblem has at most $\lceil n/2 \rceil + 1$ digits.
- We have 3 subproblems $\Longrightarrow$ at most $3 \cdot T_K(\lceil n/2 \rceil + 1)$ operations.
- 6 additions of two numbers having at most $2n$ digits.

Solving Recursion

$$T_K(n) \le \begin{cases} 3n^2 + 2n, & \text{if } n \le 3 \\ 3 \cdot T_K(\lceil n/2 \rceil + 1) + 6 \cdot 2 \cdot n, & \text{if } n \ge 4 \end{cases}$$

Claim: $T_K(n) \le 207 \cdot n^{\log 3}$

Consider $n = 2^k + 2,\ k \ge 1$ as base case

Why?

$$\lceil n/2 \rceil + 1 = \lceil (2^k + 2)/2 \rceil + 1 = 2^{k-1} + 1 + 1 = 2^{k-1} + 2$$

Proof

Claim:

$$T_K(2^k+2) \le 69\cdot 3^k - 24\cdot 2^k - 12$$

Proof (by induction):

k=0:

$$T_K(2^0+2) = T_K(3) \le 3\cdot 3^2 + 2\cdot 3 = 33 = 69\cdot 3^0 - 24\cdot 2^0 - 12$$

Proof

k>0:

$$
\begin{aligned}
T_K(2^k+2) &\le 3T_K(2^{k-1}+2)+12\cdot(2^k+2) \\
&\le 3\cdot\left(69\cdot 3^{k-1}-24\cdot 2^{k-1}-12\right)+12\cdot(2^k+2) \\
&\le 69\cdot 3^k-24\cdot 2^k-12 \qquad \square
\end{aligned}
$$

Extension to general n:

Let k be minimal integer such that $n \le 2^k+2$.

Implies: $k \le \log n + 1$

Extension to general n:

$$T_K(n) \leq 207 \cdot n^{\log 3}$$

Proof:

Multiplying n-digit integers is no more costly than multiplying $(2^k + 2)$-digit integers.

Implies: $T_K(n) \leq T_K(2^k + 2)$

We have:

$$T_K(n) \le 69 \cdot 3^k - 24 \cdot 2^k - 12$$

$$\le 207 \cdot 3^{\log n}$$

$$\le 207 \cdot n^{\log 3}$$

Using: $3^{\log n} = 2^{(\log 3)\cdot(\log n)} = n^{\log 3}$ $\square$

Comparison of Multiplication Algorithms

- Karatsuba Multiplication is asymptotically faster than School Multiplication.
- But observe that the leading constant for Karatsuba Multiplication is large.
- This means that Karatsuba multiplication is only preferable if n is large enough.

Combination of Karatsuba and School Multiplication

- Both methods multiply n digit numbers.
- We can use Karatsuba for large values of n and School multiplication for small numbers of n.

Set threshold n_0:

If $n<=n_0$: Use School Multiplication

If $n>n_0$: Use Karatsuba Multiplication

Runtime dependent of recursion threshold

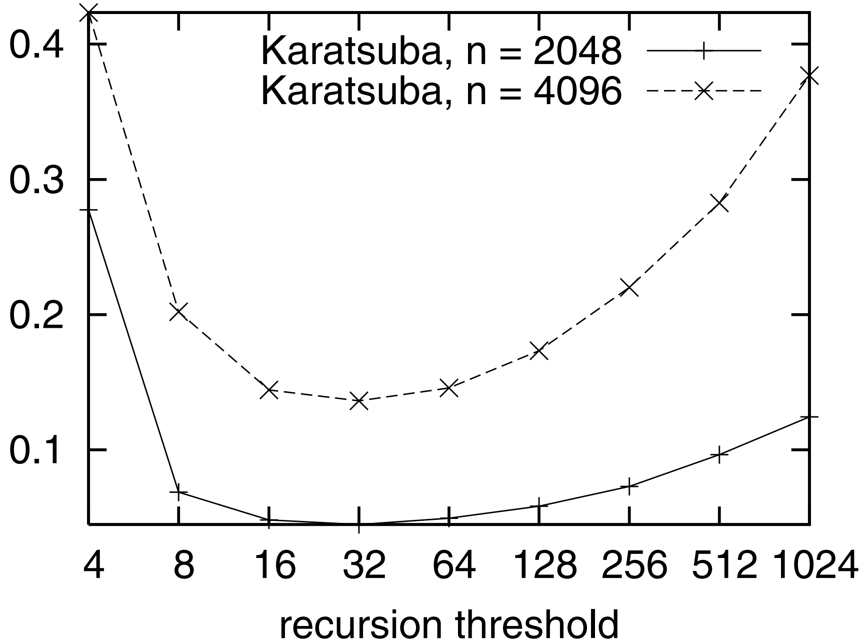

Fig. 1.4. The running time of the Karatsuba method as a function of the recursion threshold n_0. The times consumed for multiplying 2048-digit and 4096-digit integers are shown. The minimum is at $n_0 = 32$

Mehlhorn/Sanders page 12

General Tool for Solving Recursion

- Solving recursive formulas can be complicated.
- Often requires induction proofs and a good guess about what to proof.
- Would be great to have a general tool for solving standard recursive formulas.
- The master theorem provides a general way of solving recursion.

Master Theorem

Theorem (master theorem, simple form):

For positive constants a, b, c, and d, and $n = b^k$ for some integer k, consider the recurrence

$$r(n) = \begin{cases} a, & \text{if } n = 1 \\ cn + d \cdot r(n/b), & \text{if } n \geq 2 \end{cases}$$

then

$$r(n) = \begin{cases} \Theta(n), & \text{if } d < b \\ \Theta(n \log n), & \text{if } d = b \\ \Theta(n^{\log_b d}) & \text{if } d > b. \end{cases}$$

For proof see Mehlhorn/Sanders (pages 38/39)

Example Master Theorem

$$T(n) \leq \begin{cases} 1, & \text{if } n = 1 \\ 4 \cdot T(\lceil n/2 \rceil) + 3 \cdot 2 \cdot n, & \text{if } n \geq 2 \end{cases}$$

Consider $n = 2^k$

$a = 1$, $b = 2$, $c = 6$, and $d = 4$

$d > b : \Theta(n^{\log_b d}) = \Theta(n^{\log_2 d}) = \Theta(n^2)$

Summary

- Different types of algorithms for integer multiplication
- Divide and conquer is an important concept in algorithmics
- Analysis of recursive formulas
- Comparison of School and Karatsuba Multiplication.
- Master theorem is a general tool for solving standard recursive formulas.