Научно-исследовательская практика Лето 2025

Алгоритм Карацубы

Scribe: И. А. Герасимов

1 Введение

В 1960 году на семинаре А. Н. Колмогорова по математическим задачам энергетики была рассмотрена задача умножения двух чисел [1]. На тот момент основным подходом являлось умножение в столбик, обладающее асимптотической сложностью $O(n^2)$ арифметических операций [2]. Колмогоров предположил, что $O(n^2)$ представляет собой нижнюю границу сложности для любого алгоритма умножения. Однако в 1962 году А. А. Карацуба и Ю. П. Офман разработали алгоритм, опровергший данную гипотезу, снизив сложность до $O(n^{\log_2 3}) \approx O(n^{1.585})$ [1, 2]. На сегодняшний день для чисел очень большой длины существуют более эффективные методы, такие как алгоритм Шёнхаге–Штрассена [3], однако алгоритм Карацубы заложил основу для их развития и остаётся практически значимым для чисел средней длины.

2 Алгоритм

Ниже представлено формальное описание алгоритма Карацубы в виде псевдокода, основанное на оригинальной работе [1]. Алгоритм принимает два целых числа u, v и основание системы счисления sc, возвращая их произведение.

Algorithm 1 Алгоритм Карацубы для умножения

Input: Целые числа u, v, основание системы счисления $sc = 10$
Output: Произведение $u \times v$

if $u < sc$ или $v < sc$ **then**
    return $u \times v$
Определить длину u и v в системе счисления sc: len_u, len_v
$n \leftarrow \max(len_u, len_v)//2$
$B \leftarrow sc^n$
Разделить u на старшую часть a и младшую часть b: $a \leftarrow u//B$, $b \leftarrow u \mod B$
Разделить v на старшую часть c и младшую часть d: $c \leftarrow v//B$, $d \leftarrow v \mod B$
$x \leftarrow \mathrm{karN}(a, c, sc)$
$y \leftarrow \mathrm{karN}(b, d, sc)$
$z \leftarrow \mathrm{karN}(a + b, c + d, sc)$
return $x \times B^2 + (z - x - y) \times B + y$

3 Анализ

В данном разделе приведены доказательства корректности и анализ сложности алгоритма Карацубы, основанные на оригинальной статье [1] и классической литературе [2].

Теорема 1 (Корректность алгоритма Карацубы)**.** *Алгоритм Карацубы, описанный в алгоритме 1, корректно вычисляет произведение двух неотрицательных целых чисел u и v для систем счисления $sc \in \{2, 10\}$.*

Доказательство. Докажем корректность методом математической индукции по длине чисел $n = \max(\lfloor \log_{sc}(u) \rfloor + 1, \lfloor \log_{sc}(v) \rfloor + 1)$.

База индукции. Если $u < sc$ или $v < sc$, числа имеют длину не более одной цифры в системе счисления sc. Алгоритм возвращает $u \cdot v$, что тривиально корректно.

Индукционный переход. Предположим, что алгоритм корректен для чисел длиной менее n. Рассмотрим числа u и v длиной не более n. Алгоритм выполняет следующие шаги:

1. Вычисляет $n' = \max(len_u, len_v)//2$, где $len_u = \lfloor \log_{sc}(u) \rfloor + 1$ (или 1, если $u = 0$), аналогично для len_v.
2. Задаёт основание разбиения $B = sc^{n'}$.
3. Разбивает $u = a \cdot B + b$ и $v = c \cdot B + d$, где $a = u//B$, $b = u \mod B$, $c = v//B$, $d = v \mod B$.
4. Рекурсивно вычисляет $x = a \cdot c$, $y = b \cdot d$, $z = (a + b) \cdot (c + d)$.
5. Возвращает $x \cdot B^2 + (z - x - y) \cdot B + y$.

Покажем, что результат равен $u \cdot v$. Выразим произведение:

$$u \cdot v = (a \cdot B + b) \cdot (c \cdot B + d) = a \cdot c \cdot B^2 + (a \cdot d + b \cdot c) \cdot B + b \cdot d.$$

Вычислим $(z - x - y)$:

$$z = (a + b) \cdot (c + d) = a \cdot c + a \cdot d + b \cdot c + b \cdot d,$$

$$z - x - y = (a \cdot c + a \cdot d + b \cdot c + b \cdot d) - a \cdot c - b \cdot d = a \cdot d + b \cdot c.$$

Подставим в итоговое выражение:

$$x \cdot B^2 + (z - x - y) \cdot B + y = a \cdot c \cdot B^2 + (a \cdot d + b \cdot c) \cdot B + b \cdot d = u \cdot v.$$

Так как a, b, c, d имеют длину не более $\lceil n/2 \rceil < n$, по индукционному предположению x, y, z вычислены корректно. Следовательно, алгоритм возвращает правильный результат. □

Теорема 2 (Сложность алгоритма Карацубы)**.** *Алгоритм Карацубы обладает асимптотической сложностью $O(n^{\log_2 3}) \approx O(n^{1.585})$, где n — длина входных чисел в цифрах системы счисления sc [2].*

Доказательство. Обозначим $T(n)$ — время работы алгоритма для чисел длиной n. Алгоритм выполняет:

- Определение длины чисел и вычисление $B = sc^{n'}$ за $O(n)$ операций.
- Разбиение чисел на a, b, c, d за $O(n)$ операций.
- Три рекурсивных вызова для чисел длиной $\lceil n/2 \rceil$: $T(\lceil n/2 \rceil)$.
- Вычисление $a + b$, $c + d$ и $z - x - y$ за $O(n)$ операций.
- Сборка результата $x \cdot B^2 + (z - x - y) \cdot B + y$ за $O(n)$ операций.

Рекуррентное соотношение:

$$T(n) = 3T\left(\left\lceil \frac{n}{2} \right\rceil\right) + O(n).$$

Для n — степени двойки: $T(n) = 3T(n/2) + O(n)$. Применяя теорему о рекуррентных уравнениях (Master Theorem) [3], где $a = 3$, $b = 2$, $f(n) = O(n)$, $\log_b a = \log_2 3 \approx 1.585 > 1$, получаем:

$$T(n) = O(n^{\log_2 3}) \approx O(n^{1.585}).$$

Для произвольного n асимптотика сохраняется, так как добавочная константа не влияет на порядок роста. □

4 Характеристики машины

Тестирование проводилось на компьютере с процессором Intel Core i5 11-го поколения (2.40 ГГц), 8 ГБ оперативной памяти, SSD накопителем WDC PC объёмом 512 ГБ и операционной системой Windows 10. Использовалась система компьютерной алгебры SageMath версии 10.6, обеспечивающая достаточную производительность для анализа алгоритма на больших входных данных.

5 Сравнение производительности

Для оценки производительности алгоритма Карацубы были проведены эксперименты с числами длиной 50, 100, 200 и 250 цифр в десятичной системе счисления ($sc = 10$). Время выполнения измерялось функцией `timeit` в SageMath, усреднённое по 100 запускам. Результаты сравнения с встроенным умножением SageMath приведены в таблице 1 [3].

Размер входа (цифр)	Время алгоритма Карацубы (сек)	Время встроенного умножения (сек)
50	0.547	0.000000105
100	0.154	0.000000176
200	0.444	0.000000585
250	0.1100	0.000000858

Таблица 1: Сравнение времени выполнения

6 Заключение

Алгоритм Карацубы, предложенный А. А. Карацубой и Ю. П. Офманом в 1962 году [1], стал важным шагом в развитии теории алгоритмов умножения, снизив сложность с $O(n^2)$ до $O(n^{\log_2 3}) \approx O(n^{1.585})$ [2]. Эксперименты с числами длиной 50–250 цифр подтвердили его корректность, однако показали значительное отставание по времени выполнения от встроенного умножения SageMath (0.154–0.547 сек против 0.000000105–0.000000858 сек), что объясняется использованием в SageMath оптимизированных библиотек, таких как GMP, и методов на основе быстрого преобразования Фурье [3].

Алгоритм сохраняет теоретическую ценность как первый метод, опровергший гипотезу Колмогорова [1], и послужил основой для разработки более совершенных алгоритмов, например, Шёнхаге–Штрассена [3]. Он остаётся актуальным для чисел средней длины и в образовательных целях [2]. Дальнейшие исследования могут включать тестирование на больших входных данных и оптимизацию реализации.

Список литературы

[1] Карацуба А., Офман Ю. Умножение многозначных чисел на автоматах. *Доклады Академии наук СССР*, 145(2):293–294, 1962.

[2] Donald E. Knuth. *The Art of Computer Programming, Volume 2: Seminumerical Algorithms.* Addison-Wesley, 3rd edition, 1997.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. *Introduction to Algorithms.* MIT Press, 3rd edition, 2009.