

Сложность алгоритмов

А. БЕЛОВ, В. ТИХОМИРОВ

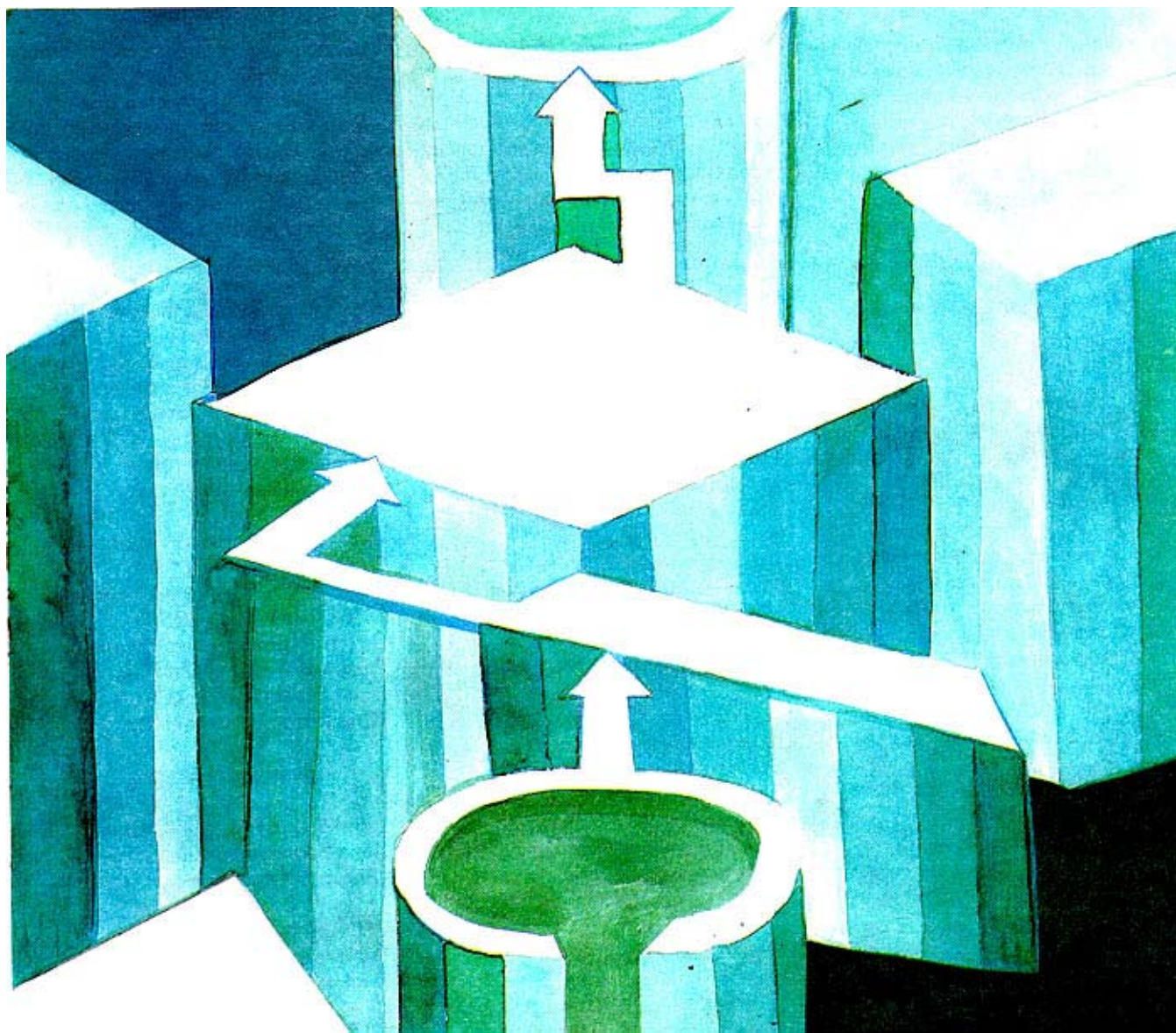


НАВЕРНОЕ, ВСЕ ПОНИМАЮТ сейчас, что такое программа, и что скорость решения задачи на компьютере зависит от того, как оно написано. В одном случае компьютер будет работать долго, выполняя большой объем вычислений, в другом – объем вычислений окажется меньше, и работа будет эффективней.

Чтобы организовать вычислительный процесс, необходимо иметь точное предписание, приводящее от исходных данных к конечному результату. Такое предписание называется *алгоритмом*. (Само слово «алгоритм» происходит от латинского написания имени великого арабского математика аль-Хорезми, жившего в VIII – IX веке.)

Время работы компьютера зависит

от количества «элементарных операций», которые ему предстоит выполнить при реализации алгоритма. И *сложность алгоритма* определяется как необходимое число таких операций. Тема «сложность алгоритмов» стала одной из самых актуальных в современной математике. В этой статье мы обсудим вопрос о сложности алгоритмов сложения и умножения чисел.



Сложение и умножение

Рассмотрим простейший алгоритм сложения двух чисел столбиком. При применении этого алгоритма, например, к числам 1018 и 1105 – когда требуется сложить тысячу восемнадцать и тысячу сто пять в десятичной системе – на вход подается пара (1018, 1105) и мы поступаем так:

$$\begin{array}{r} 1 \\ + 1018 \\ + 1105 \\ \hline 2123 \end{array}$$

Произведено шесть элементарных операций по сложению однозначных чисел: $8 + 5 = 13$ (три пишем, один «в уме»), $1 + 1 + 0 = 2$, $0 + 1 = 1$, и, наконец, $1 + 1 = 2$. А меньше четырех и нельзя: ведь хоть по разу необходимо было употребить каждую цифру чисел.

А теперь умножим друг на друга два числа (в десятичной системе):

$$\begin{array}{r} 1011 \\ \times 1101 \\ \hline 1011 \\ 0000 \\ 1011 \\ 1011 \\ \hline 1113111 \end{array}$$

Сколько операций мы совершили здесь? Элементарных умножений $4 \times 4 = 16$, и еще некоторое количество сложений. Из этого примера вытекает, что при умножении двух n -значных чисел столбиком нужно произвести n^2 элементарных умножений (n^2 раз заглянуть в таблицу умножения и заведомо не больше по порядку n^2 раз произвести элементарные сложения.)

Упражнение 1. Покажите, что алгоритм деления в столбик n -значного числа на m -значное требует по порядку величины mn элементарных операций (не менее mn и не более $10mn$).

А какова сложность процедуры умножения? Каково минимальное необходимое число элементарных операций?

Умножение столбиком известно многие столетия, едва ли еще не со времени выхода в свет «Алгебры» аль-Хорезми. Эта процедура, которую все изучали еще в начальных классах средней школы, выглядит как самый быстрый алгоритм: ведь каждая цифра одного числа должна вроде бы провзаимодействовать с каждой цифрой другого.

Тем не менее, всего тридцать семь лет назад московским математиком Анатолием Александровичем Карацубой (тогда 25-летним молодым ассистентом, а ныне он профессор МГУ, известный специалист по теории чисел) было совершено открытие, которого трудно было ожидать. Он обнаружил метод, куда более эффективный, чем умножение в столбик!

Метод Карацубы

Пусть x и y – два $2n$ -значных десятичных числа:

$$x = (x_{2n-1}, \dots, x_0), \quad y = (y_{2n-1}, \dots, y_0), \\ x_i, y_i = 0, 1, \dots, 9.$$

Представим их в виде

$$x = 10^n \xi_1 + \xi_0, \quad y = 10^n \eta_1 + \eta_0, \quad (i)$$

где ξ_i, η_i – n -значные числа. Воспользовавшись тождеством

$$(\xi_1 - \xi_0)(\eta_0 - \eta_1) = \\ = -\xi_1 \eta_1 - \xi_0 \eta_0 + \xi_1 \eta_0 + \xi_0 \eta_1, \quad (ii)$$

получаем:

$$xy = (10^n \xi_1 + \xi_0)(10^n \eta_1 + \eta_0) = \\ = (10^{2n} + 10^n) \xi_1 \eta_1 + \\ + 10^n (\xi_1 - \xi_0)(\eta_0 - \eta_1) + (10^n + 1) \xi_0 \eta_0.$$

Мы свели задачу умножения $2n$ -разрядных чисел к трем операциям для n -разрядных чисел $\xi_1 \eta_1$, $(\xi_1 - \xi_0)(\eta_0 - \eta_1)$, $\xi_0 \eta_0$ и еще к операциям сложения и сдвига.

На первый взгляд может показаться, что выигрыш по сравнению с обычным умножением незначителен – только $3/4$. Но ведь и сами n -значные числа мы будем умножать таким же образом! Поэтому множитель $3/4$ будет возникать при каждом удвоении числа разрядов. И, например, при умножении 1024-значных чисел накопится более чем десятикратный выигрыш.

Умножение Карацубы на n -значных числах будет эффективнее умножения в столбик по порядку величины в $C \cdot \left(\frac{3}{4}\right)^{\log_2 n}$ раз. Это дает оценку порядка $n^{\log_2 3} \approx n^{1.6}$ на число операций. В самом деле, пусть $2^k < n < 2^{k+1}$. Тогда $T(n)$ – число необходимых операций для умножения n -значных чисел – оценивается по порядку как $3^k = (2^k)^{\log_2 3} \approx n^{\log_2 3}$. Можно рас-

суждать и так. Из нашего построения вытекает рекуррентное неравенство

$$T(2n) \leq 3T(n) + cn,$$

из которого легко выводится асимптотическое неравенство

$$T(n) \ll n^{\log_2 3}.$$

Упражнение 2. Проведите соответствующие рассуждения аккуратно.

Продемонстрируем метод Карацубы на примере умножения чисел, с которыми мы оперировали: $1101 \cdot 1011$.

Воспользовавшись тождеством $(a_1 - a_0)(b_0 - b_1) = -a_1 b_1 - a_0 b_0 + a_1 b_0 + a_0 b_1$, представим четырехзначные числа A и B в виде $A = 10^2 a_1 + a_0$ и $B = 10^2 b_1 + b_0$, где a_0, a_1, b_0, b_1 – двузначные числа. Тогда

$$AB = (10^2 a_1 + a_0)(10^2 b_1 + b_0) = \\ = (10^4 + 10^2) a_1 b_1 + \\ + 10^2 (a_1 - a_0)(b_0 - b_1) + (10^2 + 1) a_0 b_0.$$

Видно, что достаточно произвести всего три умножения двузначных чисел $a_1 \times b_1$, $(a_1 - a_0) \times (b_0 - b_1)$ и $a_0 \times b_0$, на каждое из которых мы затратим не больше четырех элементарных умножений, т.е. всего мы затратим 12 умножений вместо 16 (и некоторое количество сложений).

В нашем случае

$$a_1 = 11, \quad a_0 = 1, \quad b_1 = 10, \quad b_0 = 11,$$

откуда

$$1101 \cdot 1011 = \\ = (10^2 \cdot 11 + 1) \cdot (10^2 \cdot 10 + 11) = \\ = (10^4 + 10^2) \cdot 11 \cdot 10 + 10^2 \cdot 10 \cdot 1 + \\ + (10^2 + 1) \cdot 1 \cdot 11 = 1113111.$$

Числа и многочлены

Первоначально метод Карацубы выглядел трюком. Но в следующем, 1963 году студент механико-математического факультета МГУ Андрей Тоом (он был участником первой международной математической олимпиады 1959 года, где получил третью премию) осознал природу этого метода и нашел его замечательное обобщение.

В чем идея метода Карацубы и Тоома? Известно, что многие свойства чисел и многочленов очень похожи. Многие проблемы, связанные с уравнениями, сперва решались для многочленов. Посмотрим на десятич-

ную запись некоторого числа x :

$$x = x_0 + x_1 10^1 + x_2 10^2 + \dots + x_n 10^n,$$

где x_i – цифры. Не правда ли, это очень похоже на запись многочлена? Да и умножению в столбик чисел соответствует умножение в столбик многочленов. (Разница только в отсутствии переноса разрядов у многочленов.)

А нельзя ли два многочлена умножить быстрее, чем в столбик, используя меньшее число умножений коэффициентов? Оказывается, можно! Пусть $P(t)$ и $Q(t)$ – два многочлена степеней m и n соответственно. Тогда их произведение $P(t)Q(t)$ есть многочлен степени $m + n$. А он определяется своими значениями в $m + n + 1$ точке. Например, в точках $0, \pm 1, \pm 2, \dots, \pm(m+n)$. Поэтому, чтобы найти многочлен PQ , ищем значения P и Q в этих точках и перемножаем (получается $m + n + 1$ умножение). Тем самым мы найдем значения PQ в $m + n + 1$ точке, по которым восстанавливается PQ и находятся все его коэффициенты.

Например, пусть $P(t) = At + B$, $Q(t) = Ct + D$ – линейные двучлены. Их произведение – квадратный трехчлен $PQ(t) = (AC)t^2 + (BC + AD)t + (BD)$. Найдем его коэффициенты.

1. Находим $BD = P(0)Q(0)$ – первое умножение.

2. Находим $(A + B)(C + D) = P(1)Q(1)$ – второе умножение.

3. Находим $(B - A)(D - C) = P(-1)Q(-1)$ – третье умножение.

Беря полуразность выражений $(A + B)(C + D)$ и $(B - A)(D - C)$, находим второй коэффициент PQ , т.е. $BC + AD$, а вычитая BD из полусуммы $(A + B)(C + D)$ и $(B - A)(D - C)$, находим AC . И мы пришли почти что к формулам Карацубы!¹

Идея метода Карацубы – Тоома заключается в том, чтобы разбить запись $2n$ -значных чисел $X = AB$, $Y = CD$ на блоки по n разрядов. Тем самым числа X и Y представляются в виде $At + B$ и $Ct + D$, где $t = 10^n$ для десятичной системы счисления (в общем случае $t = q^n$, q – основание системы счисления). А после этого можно перемножить полученные (линейные) многочлены.

Теперь ясно, как обобщить этот метод. Попробуем разбивать числа не

¹ Разница только в том, что мы брали $(B - A)(D - C)$ – значение в (-1) , а Карацуба вместо этого брал AC – коэффициент при главном члене на бесконечности.

на два, а на большее число блоков. Например, на три. Пусть $x_1 = \overline{A_1 B_1 C_1}$, $x_2 = \overline{A_2 B_2 C_2}$ – разбиение $3n$ -значных чисел x_1, x_2 на блоки по n разрядов. Пусть $t = 10^n$. Тогда числа x_1, x_2 можно представить в виде $P(t) = A_1 t^2 + B_1 t + C_1$ и $Q(t) = A_2 t^2 + B_2 t + C_2$ соответственно. Будем действовать, как в предыдущем случае. Найдем произведение многочленов P и Q . Для этого найдем их значения в точках $0, \pm 1, \pm 2$ и перемножим. Получится 5 умножений k -значных чисел (и еще несколько сложений и умножений на 2 и 4). Далее, зная значения многочлена PQ в этих точках, найдем его коэффициенты. Они получаются путем умножения и деления на небольшие числа. Это требует не более 40 сложений (см. упражнение 1).

Таким образом, мы свели задачу умножения $3n$ -разрядных чисел к пяти операциям для n -разрядных чисел и еще к операциям сложения, деления на маленькие числа и сдвига. Это дает оценку порядка $n^{\log_3 5}$ на число операций. В самом деле, пусть $3^k < n < 3^{k+1}$. Тогда $T(n)$ (число необходимых операций для умножения n -значных чисел) оценивается по порядку как

$$5^k = (3^k)^{\log_3 5} \approx n^{\log_3 5} \approx n^{1.3}.$$

Упражнения

3. Выведите формулы для коэффициентов многочлена $R(t)$ четвертой степени по его значениям в точках $0, \pm 1, \pm 2$.

4. Пусть e – фиксированное натуральное число.

а) Докажите, что алгоритм «умножения в столбик» n -значных чисел на число e имеет сложность порядка n . Его сложность можно оценить величиной $10k \cdot n$, где k – число разрядов в числе e .

б) Докажите, что алгоритм «деления в столбик» n -значных чисел на число e имеет сложность порядка n . Его сложность можно оценить величиной $10k \cdot n$, где k – число разрядов в числе e .

5. Покажите, что для нахождения произведения двух квадратных трехчленов с n -значными коэффициентами достаточно ограничиться $5n$ -значными умножениями и $30n$ элементарными операциями.

6. Докажите рекуррентное неравенство

$$T(3n) \leq 5T(n) + 30n.$$

Выведите из него асимптотическое неравенство

$$T(n) \ll n^{\log_3 5}.$$

Естественно, десятичную запись чисел можно разбивать не на три, а на большее число блоков. Проведем об-

щее рассуждение. Пусть x и y – два $n(r+1)$ -значных числа. Представим их в виде

$$x = 10^{rn} \xi_r + \dots + 10^n \xi_1 + \xi_0,$$

$$y = 10^{rm} \eta_r + \dots + 10^n \eta_1 + \eta_0$$

и положим

$$p(t) = \sum_{k=0}^r \xi_k t^k, \quad q(t) = \sum_{k=0}^r \eta_k t^k,$$

$$s(t) = \sum_{k=0}^{2r} \zeta_k t^k \Rightarrow xy = s(10^n).$$

В следующем пункте будет показано, что

Коэффициенты полинома $s(\cdot)$ вычисляются через линейные выражения от $2r + 1$ чисел $\{p(k)q(k)\}_{k=0}^{2r}$, и это требует $C(r)n$ операций.

Таким образом, нахождение произведения $n(r+1)$ -значных чисел требует $2r + 1$ «больших умножений» $(n + D_r)$ -значных чисел, где D_r – некоторая константа, зависящая от r , и $C(r)n$ элементарных операций.

Упражнение 7. Докажите, что константу D_r можно положить равной числу знаков в числе $r^{r+1} = r \cdot r^r$, $t = r$.

В итоге приходим к оценке

$$T((r+1)n) \leq (2r+1)T(n + D_r) + cn.$$

Пусть

$$(r+1)^s \leq n \leq (r+1)^{s+1}.$$

Тогда легко видеть, что при некотором α_r имеет место неравенство

$$T(n) \leq \alpha_r (2r+1)^s,$$

из которого вытекают асимптотические соотношения

$$T(n) \ll n^{\log_{r+1}(2r+1)} \ll n^{1 + \log_{r+1} 2}.$$

Поскольку для любого наперед заданного ε при достаточно большом r верно неравенство $r^\varepsilon > 3$, то $r^{1+\varepsilon} > 3r$. Теперь ясно, что можно выбрать такое r , что при достаточно больших n выполняется неравенство

$$T(n) < n^{1+\varepsilon}.$$

Мы получили такой результат:

Теорема (А.Тоом). Для любого $\varepsilon > 0$ существует такая постоянная $c(\varepsilon)$ и такой метод умножения, что число элементарных операций $T(n)$, которые необходимо выполнить, чтобы умножить два n -разрядных числа, удовлетворяет оценке

$$T(n) \leq c(\varepsilon)n^{1+\varepsilon}.$$

Чтобы доказать теорему Тоома и построить алгоритм быстрого умно-

жения, нам надо решить задачу *интерполяции*, т.е. восстановления коэффициентов многочлена по его значениям.

Интерполяционные полиномы

Каждый многочлен степени k однозначно определяется своими значениями в $k + 1$ точке. В самом деле, пусть $P(x)$ и $Q(x)$ – два многочлена степени k , совпадающие в $k + 1$ точке. Тогда их разность $P - Q$ обращается в ноль в этих точках и является многочленом степени не выше k . Но если многочлен степени не выше k имеет k нулей, то он тождественно равен нулю. Поэтому $P(x) \equiv Q(x)$. Однако нам нужно не просто доказать однозначность, а явно осуществить построение.

Построим многочлен k -й степени $R(x)$, принимающий в точках x_1, \dots, x_{k+1} значения y_1, \dots, y_{k+1} . Для этого достаточно уметь строить элементарные многочлены $R_i(x)$, принимающие значения 1 в точке x_i и 0 в остальных точках. Тогда многочлен $R(x)$ находится как сумма

$$\sum_{i=1}^{k+1} y_i R_i(x).$$

Чтобы обеспечить равенство нулю в точках $x_i, i \neq j$, рассмотрим произведение

$$q_i(x) = (x - x_1)(x - x_2) \times \dots \times \left(\overset{\wedge}{x - x_i} \right) \dots (x - x_{k+1}),$$

где $\left(\overset{\wedge}{x - x_i} \right)$ означает, что соответствующий множитель опускается.

Разделив многочлен $q_i(x)$ на его значение в точке x_i , мы получим многочлен

$$R_i(x) = \frac{(x - x_1)(x - x_2)}{(x_i - x_1)(x_i - x_2)} \times \dots \times \frac{\left(\overset{\wedge}{x - x_i} \right) \dots (x - x_{k+1})}{\left(\overset{\wedge}{x_i - x_i} \right) \dots (x_i - x_{k+1})}.$$

Окончательно имеем:

$$R(x) = \sum_{i=1}^{k+1} y_i \cdot \frac{(x - x_1)(x - x_2)}{(x_i - x_1)(x_i - x_2)} \times \dots \times \frac{\left(\overset{\wedge}{x - x_i} \right) \dots (x - x_{k+1})}{\left(\overset{\wedge}{x_i - x_i} \right) \dots (x_i - x_{k+1})}.$$

Мы получили *интерполяционную формулу Лагранжа* для многочлена.

Найдем коэффициенты a_s многочлена R и коэффициенты a_{si} многочленов R_i . Воспользовавшись формулой Виета, получим

$$\sum_{j_1 < \dots < j_s; j_\alpha \neq i} x_{j_1} x_{j_2} \dots x_{j_s} / \left((x_i - x_1) \times \dots \times (x_i - x_2) \dots (x_i - x_i) \dots (x_i - x_{k+1}) \right).$$

Тогда

$$a_s = \sum_i y_i \cdot a_{si}.$$

Итак, мы получили формулы для коэффициентов интерполяционного полинома. Они быстро усложняются с ростом k – числа блоков, на которые разбивается запись числа. Это приводит к быстрому увеличению числа «малых» умножений. С другой стороны, только увеличение k позволяет экономить «большие» умножения. Компромисс зависит от числа разрядов умножаемых чисел.

Посмотрим на это чуть более подробно. В нашем случае x_i суть числа $0, \pm 1, \dots, \pm k, k$ – число блоков, на которые разбивается запись числа, степень многочлена PQ равна $2k, y_i = P(x_i)Q(x_i)$ имеют примерно $2 \cdot n/k$ разрядов. Поэтому формулы для восстановления коэффициентов PQ можно записать в виде

$$a_s = \left(\sum_{i=1}^{2k+1} y_i a_{is} \right) / \beta,$$

где β – общее кратное чисел

$$\tau_i = (x_i - x_1)(x_i - x_2) \times \dots \times \left(\overset{\wedge}{x_i - x_i} \right) \dots (x_i - x_n).$$

Несложно убедиться в том, что все τ_i делят $(2k + 1)!$ и что числа a_{is} имеют порядок $(2k + 1)!$. Таким образом, в качестве β можно взять $(2k + 1)!$, и количество разрядов в числах a_{is} и β примерно равно $2k \cdot \log_{10}(2k)$. (Это следует из *формулы Стирлинга*: $n! \approx \sqrt{2\pi n} (n/e)^n$ при больших n .) Итак, после осуществления $2k + 1$ «большого» умножения и нахождения y_i остается найти $2k + 1$ коэффициент многочлена PQ . Каждый такой коэффициент находится как сумма $2k + 1$ слагаемого $y_i a_{is}$. Одно-единственное деление на $\beta = (2k + 1)!$ можно произвести в самом конце. Последняя операция имеет сложность порядка $2k \cdot \log_{10}(2k)$. (Деление мы осуществляем в столбик.)

Если «малые» умножения осуществлять в столбик, то получается оценка

$$T(n) \leq (2k + 1)T(n/k) + (2k + 1)^2 2n/k \cdot 2k \log_{10} k + 2nk \log_{10} k.$$

Можно действовать оптимальнее, разбив десятичную запись y_i на блоки по $2k \log_{10}(2k)$ разрядов и умножая блоки с помощью быстрого умножения. Можно получить такую оценку:

$$T(n) \leq (2k + 1)T(n/k) + (2k + 1)^2 n/k^2 \cdot T(2k \log_{10} 2k) + 2nk \log_{10} k.$$

Конечно, процедуру вычисления коэффициентов a_{is} можно оптимизировать. Ведь промежуточные вычисления для одного коэффициента можно использовать для другого.

Хотя вопросы оптимизации алгоритмов умножения весьма интересны, их более подробное обсуждение выходит за рамки данной статьи, главная цель которой – рассказать о самом факте быстрого умножения. А разработка и оптимизация соответствующих алгоритмов, вместе с их программной реализацией, может послужить темой хорошего доклада на научной конференции учащихся (впрочем, на «взрослой» конференции тоже). Поэтому мы предлагаем читателю следующую задачу:

Задача на исследование. *Разработайте алгоритмы умножения многозначных чисел. Исследуйте вопрос о числе блоков разбиения для каждого шага.*

Впоследствии были найдены более совершенные схемы (А.Тоом, С.Кук, А.Шенхаге). А.Шенхаге и Ф.Штрассен построили эффективно работающий алгоритм умножения с числом элементарных операций $\leq Cn \log n \log \log n$. Эти схемы помимо интерполяционных полиномов использовали так называемое *быстрое преобразование Фурье*. Если читатель хочет заняться предложенной задачей, то мы советуем ему обратиться к замечательной книге Д.Кнута «Искусство программирования» или написать нам по адресу kanel@mccme.ru или kanel@dnttm.ru.

Нет ничего элементарнее задачи нахождения произведения двух чисел. Этой задачей занимаются в начальной школе. Но компьютер в школе не учился! И оказалось, что для эффективного решения элементарной задачи умножения необходимы интерполяционные полиномы и довольно тонкие не вполне элементарные комбинаторные методы. Вероятно, в этом заключается одна из причин, по которым быстрое умножение было открыто сравнительно недавно.