36 Реализация арифметики эллиптических кривых

36.1 Кратная точка

При реализации криптосистем на основе эллиптических кривых используется преобразование $P \mapsto dP$, где P — точка кривой, d — целое. Это преобразование применяется при нахождении открытого ключа по личному, при выработке ЭЦП, в других случаях.

Пусть эллиптическая кривая E определена над простым полем $\mathbb{F}_p$, $p > 3$, кратность d является l-битовым числом: $d = (d_{l-1} \dots d_1 d_0)_2$.

Для нахождения кратной точки мы можем воспользоваться бинарными методами (см. § 25.4):

Справа налево	*Слева направо*
1. Установить $U \leftarrow O$, $V \leftarrow P$.	1. Установить $U \leftarrow O$.
2. Для $i = 0, 1, \dots, l-1$ выполнить:	2. Для $i = l-1, l-2, \dots, 0$ выполнить:
(1) если $d_i \neq 0$, то $U \leftarrow U + V$;	(1) $U \leftarrow 2U$;
(2) $V \leftarrow 2V$.	(2) если $d_i \neq 0$, то $U \leftarrow U + P$.
3. Вернуть U.	3. Вернуть U.

Мы можем немного оптимизировать алгоритм, отказавшись от последнего удвоения при вычислениях «справа налево», и от первого удвоения при вычислениях «слева направо». В конце концов, нам потребуется $(l-1)$ удвоений и в среднем $l/2$ сложений точек.

Нетривиальные (без O) сложения и удвоения точек задаются формулами

$$(x_1, y_1) + (x_2, y_2) = (x_3, y_3) = (\lambda^2 - x_1 - x_2, \lambda(x_1 - x_3) - y_1), \quad \lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, & x_1 \neq x_2, \\ \frac{3x_1^2 + a}{2y_1}, & (x_1, y_1) = (x_2, y_2), y_1 \neq 0. \end{cases}$$

Операции с точками задействуют следующие арифметические операции в $\mathbb{F}_p$:

операции с точками	операции в $\mathbb{F}_p$	паритетное число умножений в $\mathbb{F}_p$
удвоение	$1I + 2M + 2S$	103.6
сложение	$1I + 2M + 1S$	102.8

Здесь I — обращение в $\mathbb{F}_p$, M — умножение в $\mathbb{F}_p$, S — возведение в квадрат в $\mathbb{F}_p$. При подсчете паритетного числа умножений мы использовали распространенные оценки: $1I \approx 100M$, $1S \approx 0.8M$.

В целом нахождение кратной точки в среднем паритетно $155l$ умножениям. Далее мы покажем, как можно ускорить реализацию преобразования.

36.2 Специальные простые

При генерации кривой можно выбирать поле $\mathbb{F}_p$, над которым кривая будет определяться. Свободой выбора p можно воспользоваться и задавать специальные модули, приведение по которым будет максимально быстрым.

Специальные p вида $2^n - c$ используются в следующем алгоритме.

Алгоритм Крендала

Вход: a — неотрицательное целое, $p = 2^n - c$ (c — невелико).

Выход: $a \bmod p$.

Шаги:

1. Пока $a \geqslant 2p$:

(1) Записать a в виде $a = u + v2^n$, $u < 2^n$. При этом $a = u + v(p + c) \equiv u + vc \pmod p$. Учтем данный факт на следующем шаге.

(2) $a \leftarrow cv + u$.

2. Если $a > p$, то $a \leftarrow a - p$.

3. Возвратить a.

Сложность. На каждом проходе цикла 1 выполняется одно умножение и одно сложение в $\mathbb{Z}$ (деление на 2^n игнорируется). На шаге 2 дополнительно может потребоваться 1 вычитание. Ниже оценивается число проходов цикла 1.

Замечание 36.1. Алгоритм Крендала используется в курсе школьной математики: пусть $a = (a_{n-1} \dots a_1 a_0)_{10}$, тогда $a \bmod 9 = (a_0 + a_1 + \dots + a_{n-1}) \bmod 9$. □

Теорема 36.1. Пусть в алгоритме Крендала $c < 2^{n/2}$ и $a < 2^{2n}$. Тогда будет выполнено не более 2 проходов цикла 1.

Доказательство. Составим таблицу пред- и постусловий при выполнении проходов (контролируем случаи, когда выполняется максимальное число проходов):

проход	до	после
1	$v < 2^n$	$a \leqslant (2^{n/2} - 1)(2^n - 1) + (2^n - 1) = (2^{n/2} - 1)2^n + (2^n - 2^{n/2})$
2	$v < 2^{n/2}$	$a \leqslant (2^{n/2} - 1)(2^{n/2} - 1) + (2^n - 1) = 2(2^n - 2^{n/2}) < 2p$

□

Пример 36.1. Примеры чисел, удобных для приведения Крендала: $p = 2^{255} - 19$ (Бернштейн), $p = 2^{256} - 189$, $2^{384} - 317$, $2^{512} - 569$ (СТБ 34.101.45). □

В американском стандарте FIPS 186-2,3 используются специальные простые другого вида (они называются простыми Солинаса): $p = 2^{192} - 2^{64} - 1$, $p = 2^{224} - 2^{96} + 1$, $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$, $p = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$, $p = 2^{521} - 1$. Для таких простых приведение $x \bmod p$ может быть реализовано сдвигами и сложения фрагментов двоичного слова, представляющего x.

36.3 Умножение и возведение в квадрат столбиком

Пусть большие числа представляются в системе счисления по основанию $B = 2^\omega$, где ω — длина машинного слова. Запись

$$a = (a_{n-1} \dots a_1 a_0)_B$$

означает, что $0 \leqslant a_i < B$ и $a \sum_{i=0}^{n-1} a_i B^i$.

Рассмотрим умножение двузначных чисел $a = (a_1 a_0)_B$ и $b = (b_1 b_0)_B$:

$$ab = a_1 b_1 B^2 + (a_1 b_0 + a_0 b_1)B + a_0 b_0.$$

Для определения произведения следует выполнить четыре умножения машинных слов. Оказывается, что можно затратить даже меньше базовых умножений!

Во первых, пусть $a = b$. Тогда нам потребуется только 3 умножения:

$$a^2 = a_1^2 B^2 + 2a_1 a_0 B + a_0^2.$$

Возведение в квадрат можно распространить на общий случай.

Алгоритм Возведение в квадрат

Вход: $a = (a_{n-1} \dots a_0)_B$.

Выход: $c = a^2 = (c_{2n-1} \dots c_0)_B$.

Шаги:

1. Установить $c \leftarrow 0$.
2. Для $i = 0, 1, \ldots, n-1$:
   (1) для $j = i+1, \ldots, n-1$: $c \leftarrow c + a_i a_j B^{i+j}$.
3. Установить $c \leftarrow 2c$.
4. Для $i = 0, 1, \ldots, n-1$: $c \leftarrow c + a_i^2 B^{2i}$.
5. Возвратить c.

Сложность: $n(n+1)/2$ умножений разрядов (упр.).

Упражнение 36.1. Переписать алгоритм так, чтобы в левых частях оператора присваивания фигурировали только слова c_i или другие вспомогательные машинные слова. Использовать как можно меньше сложений машинных слов. □

Замечание 36.2. Умножение не может быть медленнее возведения в квадрат более чем в 2 раза (по числу умножений машинных слов). Действительно, $ab = ((a+b)^2 - (a-b)^2)/4$, т.е. умножить можно с помощью двух возведений в квадрат. □

36.4 Умножение Карацубы — Офмана

В 1962 году молодой советский математик А. Карацуба вместе с Ю. Офманом обратил внимание, что умножение $(a_1 a_0)_B$ на $(b_1 b_0)_B$ можно выполнить, перемножив всего три B-разрядных числа:

$$(Ba_1 + a_0)(Bb_1 + b_0) = B^2 a_1 b_1 + B\Big(a_1 b_1 + (a_1 - a_0)(b_0 - b_1) + a_0 b_0\Big) + a_0 b_0.$$

Для умножения n-разрядных чисел можно воспользоваться тем же приемом и получить следующий рекурсивный алгоритм:

Алгоритм MulKara (умножение Карацубы — Офмана)

Вход: a, b — n-разрядные числа по основанию B. Для простоты n — степень двойки: $n = 1$ или $n = 2k$, где k — целое.
Выход: $c = ab = (c_1 c_0)_{B^n}$.
Шаги:

1. Если $n = 1$, то возвратить $c \leftarrow ab$.
2. Записать $a = (a_1 a_0)_{B^k}$, $b = (b_1 b_0)_{B^k}$.
3. $c_0 \leftarrow \texttt{MulKara}(a_0, b_0)$.
4. $c_1 \leftarrow \texttt{MulKara}(a_1, b_1)$.
5. $c \leftarrow c + B^k(c_1 + c_0)$.
6. $t \leftarrow \texttt{MulKara}(|a_1 - a_0|, |b_0 - b_1|)$.
7. Если $(a_1 - a_0)(b_0 - b_1) > 0$, то $c \leftarrow c + tB^k$.
8. Если $(a_1 - a_0)(b_0 - b_1) < 0$, то $c \leftarrow c - tB^k$.
9. Возвратить c.

Сложность: $3^{\log_2 n}$ умножений разрядов (упр.).

Пример 36.2. Для умножения столбиком двух 8-разрядных чисел потребуется 64 умножения разрядов, а для умножения Карацубы — Офмана — только 27. □

Замечание 36.3. Умножение столбиком выполняется за время $O((\log a)^2)$, умножение с помощью алгоритма Карацубы — Офмана — за время $O(3^{\log_2 \log a}) = O(2^{\log_2 3 \cdot \log_2 \log a}) = O((\log a)^{\log_2 3})$, а с помощью алгоритма Шёнхаге — Штрассена — за время $O(\log a \cdot \log\log a \cdot \log\log\log a)$. □

36.5 Проективные координаты

Выберем натуральные n, m и поставим в соответствие аффинной точке (x, y), $x, y \in \mathbb{F}_p$, *проективную точку*

$$\{(X : Y : Z) \colon X, Y, Z \in \mathbb{F}_p, Z \neq 0, X/Z^n = x, Y/Z^m = y\} = \{(x\lambda^n, y\lambda^m, \lambda) \colon \lambda \in \mathbb{F}_p^*\}.$$

Проективная точка представляет собой целое множество (!), простейшим представителем которого является элемент $(x, y, 1)$.

Уравнение E для аффинных точек можно преобразовать в уравнение для проективных точек. Для этого выполним замену (x, y) на $(X/Z^n, Y/Z^m)$, приведем члены к общему знаменателю и перейдем к равенству в числителе.

Полученному уравнению удовлетворяют либо не удовлетворяют сразу все элементы $(X : Y : Z)$. Поэтому можно говорить, что кривая E задана в *проективных координатах*, т. е. для проективных точек, в проективной плоскости.

При переходе к проективным координатам бесконечно удаленной точке O ставятся в соответствие проективные точки вида $(X : Y : 0)$, которые удовлетворяют E. Данные точки также называются бесконечно удаленными.

При выборе различных (a, b) получаются различные уравнения E в проективной плоскости: уравнение в стандартных проективных координатах ($n = m = 1$), уравнение в координатах Лопеса — Дахаба ($n = 1$, $m = 2$) и др.

Далее мы будем использовать *якобиановы координаты*: $n = 2$, $m = 3$. Уравнение кривой в этих координатах:

$$(Y/Z^3)^2 = (X/Z^2)^3 + a(X/Z^2) + b \Rightarrow Y^2 = X^3 + aXZ^4 + bZ^6.$$

Точка на бесконечности: $(1 : 1 : 0)$. Обратной к $(X : Y : Z)$ является точка $(X : -Y : Z)$.

От аффинной точки (x, y) можно перейти к проективной точке $(x : y : 1)$, вести вычисления в якобиановых координатах, а затем вернуться от проективной точки $(X : Y : Z) \neq (1 : 1 : 0)$ к аффинной точке $(X/Z^2, Y/Z^3)$.

Упражнение 36.2. Сложность обратного перехода: $1S + 3M + 1I$. Действительно, его можно реализовать следующим образом: $A \leftarrow Z^{-1}$, $B \leftarrow A^2$, $x \leftarrow X \cdot B$, $C \leftarrow A \cdot B$, $y \leftarrow Y \cdot C$. □

Укажем формулы сложения и удвоения точек в якобиановых координатах.

Удвоение. $P = (X_1 : Y_1 : Z_1)$, $P \neq -P$. Найдем $2P = (X_3 : Y_3 : Z_3) = (X_3' : Y_3' : 1)$.

Запишем $P = (X_1/Z_1^2 : Y_1/Z_1^3 : 1)$ и применим правило удвоения аффинной точки:

$$X_3' = \left(\frac{3X_1^2/Z_1^4 + a}{2Y_1/Z_1^3}\right)^2 - 2X_1/Z_1^2 = \frac{(3X_1^2 + aZ_1^4)^2 - 8X_1Y_1^2}{4Y_1^2Z_1^2},$$

$$Y_3' = \left(\frac{3X_1^2/Z_1^4 + a}{2Y_1/Z_1^3}\right)\left(X_1/Z_1^2 - X_3'\right) - \frac{Y_1}{Z_1^3} = \frac{3X_1^2 + aZ_1^4}{2Y_1Z_1}\left(X_1/Z_1^2 - X_3'\right) - Y_1/Z_1^3.$$

Для того, чтобы избавиться от знаменателей, выполним замену $X_3 = X_3'Z_3^2$, $Y_3 = Y_3'Z_3^3$, где $Z_3 = 2Y_1Z_1$. Координаты точки $2P = (X_3 : Y_3 : Z_3)$ примут вид:

$$\begin{aligned} X_3 &= (3X_1^2 + aZ_1^4)^2 - 8X_1Y_1^2, \\ Y_3 &= (3X_1^2 + aZ_1^4)(4X_1Y_1^2 - X_3) - 8Y_1^4, \\ Z_3 &= 2Y_1Z_1. \end{aligned}$$

Расчет координат можно выполнить за время $4M + 6S$:

$$\begin{aligned} &A \leftarrow Y_1^2, \quad B \leftarrow 4X_1A, \quad C \leftarrow 8A^2, \quad D \leftarrow 3X_1^2 + aZ_1^4, \\ &X_3 \leftarrow D^2 - 2B, \quad Y_3 \leftarrow D(B - X_3) - C, \quad Z_3 \leftarrow 2Y_1Z_1. \end{aligned}$$

Сложение точек. Пусть $P = (X_1 : Y_1 : Z_1)$, $Q = (X_2 : Y_2 : 1) \neq \pm P$. Найдем $R = P + Q = (X_3 : Y_3 : Z_3) = (X_3' : Y_3' : 1)$.

Запишем $P = (X_1/Z_1^2 : Y_1/Z_1^3 : 1)$ и применим правило сложения аффинных точек:

$$X_3' = \left(\frac{Y_2 - Y_1/Z_1^3}{X_2 - X_1/Z_1^2}\right)^2 - X_1/Z_1^2 - X_2 = \left(\frac{Y_2Z_1^3 - Y_1}{(X_2Z_1^2 - X_1)Z_1}\right)^2 - X_1/Z_1^2 - X_2,$$

$$Y_3' = \left(\frac{Y_2 - Y_1/Z_1^3}{X_2 - X_1/Z_1^2}\right)(X_1/Z_1^2 - X_3') - \frac{Y_1}{Z_1^3} = \left(\frac{Y_2Z_1^3 - Y_1}{(X_2Z_1^2 - X_1)Z_1}\right)(X_1/Z_1^2 - X_3') - Y_1/Z_1^3.$$

Для того, чтобы избавиться от знаменателей, выполним замену $X_3 = X_3'Z_3^2$, $Y_3 = Y_3'Z_3^3$, где $Z_3 = (X_2Z_1^2 - X_1)Z_1$. Координаты точки $R = (X_3 : Y_3 : Z_3)$ примут вид:

$$\begin{aligned} X_3 &= (Y_2Z_1^3 - Y_1)^2 - (X_2Z_1^2 - X_1)^2(X_1 + X_2Z_1^2), \\ Y_3 &= (Y_2Z_1^3 - Y_1)(X_1(X_2Z_1^2 - X_1)^2 - X_3) - Y_1(X_2Z_1^2 - X_1)^3, \\ Z_3 &= (X_2Z_1^2 - X_1)Z_1. \end{aligned}$$

Расчет координат можно выполнить за время $8M + 3S$:

$$\begin{gathered} A \leftarrow Z_1^2, \quad B \leftarrow Z_1A, \quad C \leftarrow X_2A, \quad D \leftarrow Y_2A, \quad E \leftarrow C - X_1, \\ F \leftarrow D - Y_1, \quad G \leftarrow E^2, \quad H \leftarrow GE, \quad I \leftarrow X_1G, \\ X_3 \leftarrow F^2 - (H + 2I), \quad Y_3 \leftarrow F(I - X_3) - Y_1H, \quad Z_3 \leftarrow Z_1E. \end{gathered}$$

В следующей таблице представлена сложность операций с точками в различных координатах. Отдельно выделен случай $a = -3$. В этом случае при удвоении точек выражение

$$3X_1^2 + aZ_1^4 = 3(X_1 - Z_1^2)(X_1 + Z_1^2)$$

можно вычислить со сложностью $1S + 1M$ (вместо $3S + 1M$).

Координаты	Сложение точек	Удвоение точек
кривая Вейерштрасса		
стандартные проективные	$12M + 3S$	$7M + 3S$
якобиановы	$12M + 4S$	$4M + 6S$
якобиановы ($a = -3$)	$12M + 4S$	$4M + 4S$
кривая Эдвардса		
Обратные	$9M + 1S$	$3M + 4S$
Проективные	$10M + 1S$	$3M + 4S$

Еще одной возможностью ускорения операций с точками является переход к другому уравнению кривой. На сегодняшний день максимальное быстродействие достигается на кривых Эдвардса, заданных уравнением

$$x^2 + y^2 = c^2(1 + dx^2y^2), \quad c, d \in \mathbb{F}_p.$$

В таблице представлена сложность сложения и удвоения на кривых Эдвардса при использовании обратных ($x = Z/X, y = Z/Y$) и проективных ($x = X/Z, y = Y/Z$) координат.

Очень подробная информация по различным уравнениям кривых и соответствующих правилах сложения представлена на Интернет-ресурсе `http://hyperelliptic.org/EFD/`.

36.6 Аддитивные цепочки

Определение 36.1. *Аддитивной цепочкой* для числа d называется последовательность натуральных чисел $(a_0 = 1, a_2, a_3, \ldots, a_n = d)$, в которой каждый следующий член получается путем сложения каких-либо двух предыдущих:

$$a_i = a_j + a_k, \quad j, k < i, \quad i = 1, 2, \ldots, n.$$

Число n называется *длиной* цепочки. □

С помощью аддитивной цепочки для d длины n можно находить d-кратное, выполняя n сложений.

Алгоритм Кратная точка с помощью аддитивных цепочек

Вход: P, $(a_0, a_1, \ldots, a_n = d)$ — аддитивная цепочка.

Выход: dP.

Шаги:

1. $P_0 \leftarrow P$.
2. Для $i = 1, \ldots, n$:
   (1) найти k и j такие, что $a_i = a_j + a_k$;
   (2) $P_i \leftarrow P_j + P_k$.
3. Возвратить P_n.

Алгоритм выполняется на памяти, позволяющей разместить n точек кривой. Для цепочек специальной структуры можно обойтись памятью меньшего объема.

Пример 36.3. В рассмотренных выше методах, основанных на двоичном разложении кратности, также использовались аддитивные цепочки. Например, для $d = 55$ аддитивные цепочки имеют следующий вид:

справа налево	*слева направо*
1, 2, 3, 4, 7, 8, 16, 23, 32, 55 (степени двойки — в переменной V)	1, 2, 3, 6, 12, 13, 26, 27, 54, 55

Для вычислений требуется память для хранения 2 (справа налево) или 1 (слева направо) дополнительной точки. □

Для ускорения вычисления кратной точки важно научиться находить аддитивные цепочки минимальной длины. Но задача нахождения таких цепочек считается трудной. Косвенным аргументом этого является следующий результат.

Теорема 36.2 (Downey, Leong, Sethi, 1981). Пусть L — язык, составленный из слов $(b_1, b_2, \ldots, b_m, l)$ таких, что существует аддитивная цепочка длины l, которая включает $b_1, \ldots, b_m$. Тогда $L \in \mathbf{NPC}$.

Пусть $l(d)$ — минимальная длина цепочки для d. Известна оценка (Эрдёш, 1960):

$$l(d) = \log_2 d + (1 + o(1))\frac{\log_2 d}{\log_2 \log_2 d}$$

(для всех достаточно больших d).

36.7 Цепочки Брауэра

Брауер (1939) предложил строить аддитивные цепочки, используя представление d не в двоичной, а в B-ичной системе счисления, где $b = 2^w$.

Цепочка Брауера для d определяется рекурсивно следующим образом:

$$Br(d) = \begin{cases} 1, 2, 3, \ldots, B-1, & d < B \text{ (цепочка может не заканчиваться на } d!), \\ Br(d'), 2d', 4d', \ldots, Bd', d, & d \geqslant B, d' = (d_{s-1} \ldots d_1)_B. \end{cases}$$

Последовательность $Br(d'), 2d', 4d', \ldots, Bd', d$ действительно является цепочкой:

1) $d_0 = d - Bd' \in \{0, 1, \ldots, B-1\}$;

2) ненулевое d_0 обязательно содержится в первых элементах цепочки.

Замечание 36.4. Каждый новый элемент цепочки Брауэра является суммой предыдущего элемента и еще некоторого элемента (возможно снова предыдущего). Цепочки с таким свойством часто называют цепочками Брауэра. □

Пример 36.4. Пусть $d = 55 = (110111)_2$, $w = 2$. Тогда $Br(d) = (Br(13), 26, 52, 55) = (Br(3), 6, 12, 13, 26, 52, 55) = (1, 2, 3, 6, 12, 13, 26, 52, 55)$. □

Цепочки Брауера могут быть преобразованы в следующий алгоритм, который часто называется *оконным методом*.

Алгоритм Оконный метод

Вход: P, $d = (d_{s-1} \dots d_1 d_0)_B$, $b = 2^w$ $(sw = l)$.

Выход: dP.

Шаги:

1. $P_0 \leftarrow 0$, $P_1 \leftarrow P$.
2. Для $i = 2, \dots, 2^w - 1$: $P_i \leftarrow P_{i-1} + P_1$.
3. $U \leftarrow 0$.
4. Для $i = s - 1, \dots, 0$:
   (1) $U \leftarrow 2^w U$ (w удвоений);
   (2) $U \leftarrow U + P_{d_i}$.
5. Возвратить U.

Сложность: $(2^k - 2)A + l/w(wD + 1A) = lD + (2^w - 2 + l/w)A$.

Установлено, что оптимальным является выбор w в окрестности $\log_2 \log_2 d - 2 \log_2 \log_2 \log_2 d$. В этом случае длина цепочки Брауэра имеет такую же асимптотику, как и оценка Эрдёша. Это впрочем не означает, что цепочки Брауера имеют минимальную длину $l(d)$.

Недостатком алгоритма является необходимость использования дополнительной памяти для хранения $2^w - 2$ точек P_i. Можно дополнительно хранить только $2^{w-1} - 1$ точек.

Алгоритм Метод скользящего окна

Параметр: w — длина окна.

Вход: P, $d = (d_{l-1} \dots d_1 d_0)_2$.

Выход: dP.

Шаги:

1. $P_0 \leftarrow 0$, $P_1 \leftarrow P$, $P_2 \leftarrow 2P$.
2. Для $i = 1, \dots, 2^{w-1} - 1$: $P_{2i+1} \leftarrow P_{2i-1} + P_2$.
3. $U \leftarrow 0$.
4. $i \leftarrow l$.
5. Пока $i \geqslant 1$:
   (1) если $d_{i-1} = 0$, то $U \leftarrow 2U$ и $i \leftarrow i - 1$;
   (2) иначе $(d_{i-1} = 0)$:
      i. найти самую длинную последовательность $d_{i-1} d_{i-2} \dots d_k$ такую, что $i - k \leqslant w$ и $d_k = 1$;
      ii. $U \leftarrow 2^{i-k} U$ ($i - k$ удвоений);

iii. $U \leftarrow U + P_{(d_{i-1}d_{i-2}\ldots d_k)_2}$;
iv. $i \leftarrow k$.

6. Возвратить U.

36.8 Аддитивно-субтрактивные цепочки

Для аффинной (или даже проективной) точки $P = (x, y)$ вычислительно легко определить обратную точку $-P$. Поэтому при вычислении кратной точки удобно задействовать не только операцию сложения, но и операцию вычитания.

Определение 36.2. *Аддитивно-субтрактивной цепочкой* для числа d называется последовательность натуральных чисел $(a_0 = 1, a_1, a_2, \ldots, a_n = d)$, в которой каждый следующий член получается путем сложения или вычитания каких-либо двух предыдущих:

$$a_i = a_j \pm a_k, \quad j, k < i, \quad i = 1, 2, \ldots, n.$$

Алгоритм построения аддитивно-субтрактивных цепочек может быть преобразован в алгоритм нахождения кратной точки. Длина аддитивно-субтрактивной цепочки может быть меньше длины аддитивной и мы получим ускорение при вычислении dP.

Рассмотрим один распространенный способ построения аддитивно-субтрактивных цепочек, основанный на использовании двоичного знакового представления.

Определение 36.3. *Знаковой* (*двоичной знаковой*) формой числа d называется слово $d_{l-1}\ldots d_1 d_0$ в алфавите $\{0, 1, -1\}$ такое, что

$$d = \sum_{i=0}^{l-1} d_i 2^i.$$

Символ -1 в знаковой форме будем обозначать через $\bar{1}$.

По знаковой форме $d_{l-1}\ldots d_1 d_0$, $d_{l-1} = 1$, можно следующим образом построить цепочку c:

1. $a \leftarrow 1$, $c \leftarrow (a)$.
2. Для $i = l-2, \ldots, 0$:
   (1) $a \leftarrow 2a$, $c \leftarrow (c, a)$;
   (2) если $d_i = 1$, то $a \leftarrow a + 1$, $c \leftarrow (c, a)$;
   (3) если $d_i = -1$, то $a \leftarrow a - 1$, $c \leftarrow (c, a)$.
3. Возвратить c.

Пример 36.5. Число $55 = (110111)_2$ имеет следующие знаковые представления:

$$55 = 10\bar{1}100\bar{1} \quad (55 = 64 - 16 + 8 - 1),$$
$$55 = 100\bar{1}00\bar{1} \quad (55 = 64 - 8 - 1). \qquad \square$$

Как видим, знаковых представлений может быть несколько. Этим представлениям соответствуют следующие цепочки:

$$(1, 2, 4, 3, 6, 7, 14, 28, 56, 55), \quad (1, 2, 4, 8, 7, 14, 28, 56, 55).$$

Определение 36.4. *Простой* знаковой формой числа $d = (d_{l-1}\ldots d_1 d_0)_2$ называется слово, полученное последовательной заменой (справа налево) в $0d_{l-1}\ldots d_1 d_0$ подстрок 01^b, $b \geqslant 1$, на подстроки $10^{b-1}\bar{1}$. $\square$

Пример 36.6. Двоичная форма числа 55 имеет вид $(110111)_2$. В слове $0110111 = 011 \parallel 0111$ подслово 0111 меняется на $100\bar{1}$, а подслово 011 — на $10\bar{1}$. Получается простая знаковая форма $55 = 10\bar{1}100\bar{1}$. $\square$

Теорема 36.3. Пусть d — случайное число из $\mathbb{Z}_{2^l}$, $l \geqslant 2$. Тогда простое знаковое представление d в среднем содержит

$$\frac{3l}{8} + \frac{1}{4}$$

ненулевых символа.

36.9 Оптимальная знаковая форма (NAF)

В простой знаковой форме могут встречаться пары последовательных символов $\bar{1}1$. Это произойдет при преобразовании $01^a01^b \mapsto 10^{a-1}\bar{1}10^{b-1}\bar{1}$. Но внутреннюю пару $\bar{1}1$ можно заменить на $0\bar{1}$, сократив тем самым число ненулевых элементов в знаковом представлении: $01^a01^b \mapsto 10^a\bar{1}0^{b-1}\bar{1}$.

Определение 36.5. *Оптимальной знаковой формой* называется знаковая форма, полученная из простой последовательной заменой справа налево сначала пар соседних символов $\bar{1}1$ на пары $0\bar{1}$, затем троек соседних символов $0\bar{1}\bar{1}$ на тройки $\bar{1}01$ и наконец пары $1\bar{1}$ на символ 1. Часто для обозначения оптимальной знаковой формы используют аббревиатуру NAF (от non-adjacent form). □

Пример 36.7. Форма $10\bar{1}100\bar{1}$ является простой знаковой для $d = 55$, а форма $100\bar{1}00\bar{1}$ — оптимальной. □

Свойства NAF:

1) NAF(d) — единственная знаковая форма, которая не содержит пар последовательных ненулевых символов;

2) NAF(d) содержит минимальное число ненулевых символов среди всех знаковых форм d;

3) если d — случайное число из $\mathbb{Z}_{2^l}$, то NAF(d) в среднем содержит $\approx l/3$ ненулевых символа (доказательство этого факта напоминает доказательство предыдущей теоремы, но намного более громоздкое).

Оптимальную знаковую форму можно построить с помощью следующего алгоритма:

Алгоритм NAF

Вход: d.

Выход: NAF(d).

Шаги:

1. $i \leftarrow 0$.
2. Пока $d \geqslant 1$:
   (1) если d — нечетное, то $d_i \leftarrow 2 - (d \bmod 4)$, $d \leftarrow d - d_i$;
   (2) если d — четное, то $d_i \leftarrow 0$;
   (3) $d \leftarrow d/2$, $i \leftarrow i + 1$.
3. Возвратить $d_{i-1}d_{i-2}\ldots d_1d_0$.

Корректность. По построению, входное d равняется $\sum_{i=0}^{l} d_i 2^i$ (d_l — последний символ NAF(d)). Очередной символ d_i определяется так, что при $d_i \in \{1, \bar{1}\}$ следующий символ будет 0 (упр.).

Алгоритм Кратная точка с помощью NAF

Вход: P, d.

Выход: dP.

Шаги:

1. $d_l \ldots d_1d_0 \leftarrow$ NAF(d).
2. $U \leftarrow O$.
3. Для $i = l, l-1, \ldots, 0$:
   (1) $U \leftarrow 2U$.
   (2) если $d_i = 1$, то $U \leftarrow U + P$.

(3) если $d_i = \bar{1}$, то $U \leftarrow U - P$.

4. Возвратить U.

Пример 36.8. Подведем итоги. Пусть вычисление кратной точки ведется в якобиановых координатах, выбран коэффициент $a = -3$, используется NAF. Тогда потребуется $l+1$ удвоений точек, в среднем $l/3$ сложений и еще вычисления $1I + 3M + 1S$ для возврата от якобиановых координат к аффинным. Общее время работы:

$$(l+1)(4M+4S) + l/3(12M+4S) + (1I+3M+1S) \approx (12.27l + 111)M.$$

36.10 Трюк Шамира

Предположим, что нам надо определить сумму двух кратных точек $dP + eQ$. Вычисление такой суммы требуется при проверке подписи ЭльГамаля или Шнорра.

Мы можем воспользоваться одним из описанных выше алгоритмов, найти сначала dP, затем eQ и наконец искомую сумму. Если мы будем использовать бинарные методы нахождения кратной точки, то нам потребуется $2l$ удвоений и в среднем l сложений.

Однако мы можем организовать вычисления на основе бинарных методов намного быстрее.

Алгоритм Шамира

Вход: P, Q, $d = (d_{l-1} \dots d_1 d_0)_2$, $e = (e_{l-1} \dots e_1 e_0)_2$.

Выход: $dP + eQ$.

Шаги:

1. $R \leftarrow P + Q$.

2. $U \leftarrow 0$.

3. Для $i = l-1, \dots, 1, 0$:

   (1) $U \leftarrow 2U$;

   (2) если $(d_i, e_i) = (1, 0)$, то $U \leftarrow U + P$;

   (3) если $(d_i, e_i) = (0, 1)$, то $U \leftarrow U + Q$;

   (4) если $(d_i, e_i) = (1, 1)$, то $U \leftarrow U + R$.

4. Возвратить U.

Сложность. Потребуется l удвоений, 1 сложение (на шаге 1) и еще в среднем $3l/4$ сложений на шагах 3.1–3.3.

Алгоритм Шамира может базироваться не обязательно на бинарных методах, могут быть использованы и другие рассмотренные выше методы определения кратной точки.

Упражнение 36.3. Разработать алгоритм определения взвешенной суммы $d_1P_1 + d_2P_2 + \dots + d_kP_k$, $d_i < 2^l$.□