МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
МОСКОВСКИЙ ЭНЕРГЕТИЧЕСКИЙ ИНСТИТУТ
(Технический университет)

А.А. Болотов, С.Б. Гашков, А.Б. Фролов,
А.А. Часовских

АЛГОРИТМИЧЕСКИЕ ОСНОВЫ ЭЛЛИПТИЧЕСКОЙ КРИПТОГРАФИИ

Москва 2000

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
МОСКОВСКИЙ ЭНЕРГЕТИЧЕСКИЙ ИНСТИТУТ
(Технический университет)

А.А.Болотов, С.Б.Гашков, А.Б.Фролов,
А.А.Часовских

АЛГОРИТМИЧЕСКИЕ ОСНОВЫ ЭЛЛИПТИЧЕСКОЙ КРИПТОГРАФИИ

Москва 2000

УДК 519.1 **Алгоритмические основы эллиптической криптографии** А.А.Болотов, С.Б.Гашков, А.Б.Фролов, А.А.Часовских. Москва: Мэи, 2000. - 100 стр.

Учебное пособие посвящено перспективному направлению криптографии с открытым ключом - криптографии эллиптических кривых.

В нем отражен опыт работы научного семинара, работавшего в 1998/99 учебном году в МЭИ под руководством А.А.Болотова и исследовавшего вопросы эффективной реализации операций в конечных полях и группах точек эллиптических кривых для имплементации эллиптической криптографии и ее приложений.

Пособие предназначено для студентов АВТФ, специализирующихся по вопросам защиты информации в вычислительных сетях

Зав. кафедрой Математического моделирования
д.ф.-м.н., проф. Ю.А.Дубинский

Посвящается Арто Саломаа,
математику, криптографу и другу

Предисловие

В последней четверти двадцатого столетия на стыке теории сложности алгоритмов, алгоритмической теории чисел и компьютерной алгебры зародилось и в наши дни переживает настоящий бум по многим аспектам направление, известное сейчас как криптография с открытым ключом, и позволившее успешно решать разнообразные задачи, связанные с зашитой информации в компьютерных сетях ([1, 14]). В последние годы интенсивно развивается криптография эллиптических кривых (эллиптическая криптография) ([15, 16]), где роль основной криптографической функции выполняет операция вычисления кратного точки эллиптической кривой, т.е. операция умножения точки эллиптической кривой на скаляр на основе операции сложения точек кривой. Последняя, в свою очередь, реализуется с использованием операций умножения, возведения в степень и инвертирования многочленов в поле Галуа $GF(2^n)$.

Особый интерес к эллиптической криптографии обусловлен теми преимуществами, которые дает ее использование в беспроводных коммуникациях,– быстродействие и небольшая длина ключа.

Например, в построенных на основе эллиптических кривых криптосистемах бинарной размерности в диапазоне от 150 до 350 обеспечивается уровень криптографической стойкости, который требует использования в известных криптосистемах бинарной размерности от 600 до 1400 и более.

Настоящее учебное пособие и посвящено этому современному направлению криптографии с открытым ключом - криптографии эллиптических кривых.

В нём отражен опыт работы научного семинара на кафедре математического моделирования МЭИ под руководством

А.А.Болотова, на котором исследовались вопросы эффективной реализации арифметических операций в конечных полях и группах точек эллиптических кривых, а также проблемы имплементации криптографических протоколов эллиптической криптографии и ее приложений к защите электронной информации. При его подготовке использованы научные статьи авторов [2, 3, 4, 5, 6, 7], а также другие источники, приведенные в списке литературы.

Наряду с фундаментальными алгебраическими аспектами криптографии эллиптических кривых особое внимание уделяется вопросам эффективной реализации основных операций и основанных на них криптографических протоколов с учетом особенностей и возможностей компьютера, то есть методам их имплементации.

Учебное пособие состоит из настоящего предисловия, четырех глав и двух приложений.

Первая глава посвящена обсуждению основных понятий и свойств конечных полей и их расширений, а также методам построения и анализа неприводимых многочленов.

Во второй главе рассматриваются эффективные алгоритмы умножения, возведения в степень и инвертирования многочленов в полях Галуа $GF(2^n)$ и способы их компьютерной реализации.

Третья глава посвящена эллиптическим кривым и используемым в криптографии их свойствам. Описываются операции в группе точек и рассматриваются особенности кривых над полем характеристики 2. Обсуждаются результаты компьютерных экспериментов по реализации операций в конечных полях и абелевой группе точек эллиптической кривой.

В четвертой главе рассматриваются некоторые криптографические протоколы, реализуемые в двух вариантах - с использованием операции умножения и возведения в степень в конечном поле и с использованием операции группы точек эллиптической кривой и умножения на скаляр точки такой кривой.

В двух приложениях приведены таблицы неприводимых многочленов с тремя и пятью слагаемыми.

Глава 1

Конечные поля

1.1 Поля

1.1.1 Основные понятия

Поле и расширения поля

Напомним основные определения и свойства поля.

Полем называется множество $\mathcal{F}$ с операциями *сложения* и *умножения,* которые удовлетворяют ассоциативному, коммутативному и дистрибутивному законам, причём имеются как аддитивная (0), так и мультипликативная (1) единицы, каждый элемент имеет обратный элемент по сложению , кроме того каждый элемент, кроме аддитивной единицы 0 имеет и обратный элемент по умножению.

Примерами являются Q – поле рациональных чисел, R – поле действительных чисел, C – поле комплексных чисел,

Поле $\mathcal{K}$, такое, что $\mathcal{F} \subset \mathcal{K}$, называется расширением поля $\mathcal{F}$, например, поле C есть расширение как поля Q, так и поля R, последнее является расширением поля Q. Число k элементов поля называется *порядком* поля. Различают бесконечные поля (например, множество рациональных чисел) и конечные поля, например, поле {0,1} с операциями сложения по модулю два и умножения (см. пример). Конечные поля называются *полями Галуа* . Поле Галуа порядка k обозначается $GF(k)$ или $\mathcal{F}_k$.

Пример 1.1. Простейшим конечным полем является бинарное поле $GF(2)$ с операциями $\oplus$ сложения по модулю 2 и $\cdot$ умножения.

Эти операции определяются таблицами

$\oplus$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Пример 1.2. Рассмотрим отношение конгруэнтности (сравнимости) по модулю данного числа m на расширенном (включающем число 0) множестве натуральных чисел $\mathbf{N}^+$ (см. п. 1.1). Это отношение является отношением эквивалентности и разбивает множество $\mathbf{N}^+$ на классы эквивалентности, или смежные классы, по модулю m. В качестве обозначений этих классов можно взять наименьшие числа классов. Определим на этих числах операции сложения и умножения по модулю m.

Теорема 1.1. *Множество смежных классов по модулю* m (*или их обозначений*) *с операциями сложения и умножения по модулю* m *на множестве обозначений этих классов является полем тогда и только тогда, когда* $m = p$, *где* p – *простое число. Единицами по сложению и умножению этого поля* $GF(p)$ *являются классы, содержащие числа* 0 *и* 1 *соответственно.*

Порядком элемента называется наименьший положительный показатель его степени, равной 1.

Элемент g поля называется *примитивным,* или *образующим,* если для любого другого ненулевого элемента a поля найдется неотрицательное число x, такое, что $a = g^x$. Как видим, образующими конечного поля $\mathcal{F}_k$ являются элементы порядка $k - 1$.

Рассмотренное поле классов конгруэнтности целых чисел по модулю простого числа p $GF(p)$) (обозначается также Z/pZ или $\mathcal{F}_p$) и называется простым полем.

Если многократное сложение 1 не позволяет получить 0, то поле называется полем характеристики ноль, в этом случае оно содержит копию поля рациональных чисел. В противном случае, если существует простое число p такое, что p-кратное сложение 1 даёт 0, число p называется характеристикой поля. В этом случае поле содержит копию поля Z/pZ.

Кольцо многочленов, поля разложения многочленов, алгебраические замыкания

Кольцо многочленов над полем $\mathcal{F}$ во множестве переменных $X = \{X_1, \ldots, X_m\}$, обозначается $\mathcal{F}[X]$, образуется как всевозможные суммы произведений степеней этих переменных с коэффициентами из $\mathcal{F}$. (Напомним, что в отличие от поля не каждый отличный от 0 элемент кольца имеет обратный элемент по умножению). Полиномы в $\mathcal{F}[X]$ складываются и умножаются по тем же правилам, что и многочлены над действительными переменными.

Говорят, что многочлен g делит многочлен f, если существует многочлен h такой, что $f = g \times h$, $f, g, h \in \mathcal{F}[X]$.

Наибольшим общим делителем (н.о.д.) двух многочленов g, f кольца $\mathcal{F}[X]$ называется нормированный многочлен наибольшей степени, который делит каждый из этих многочленов.

Многочлен $f \in \mathcal{F}[X]$ называется неприводимым, если $f = g \times h$ только в случае, когда либо g либо h является константой. Неприводимые многочлены среди многочленов играют ту же роль, что простые числа среди целых чисел. Далее будут рассматриваться только многочлены одной переменой (случай $m = 1$, $X = X_1$). Степенью многочлена называется наибольшая степень d переменной X, присутствующая в многочлене с ненулевым коэффициентом a_d. При этом многочлен называется нормированным, Если $a_d = 1$.

Всякое кольцо многочленов одной или более переменных имеет единственную факторизацию, в том смысле, что каждый многочлен из кольца может быть представлен в виде произведения неприводимых многочленов единственным с точностью до порядка множителей образом.

Производная многочлена f степени d одной переменной X определяется по правилу nX^{n-1} для составляющих многочлен степеней X^n переменной X, а не через понятие предела ввиду отсутствия понятия расстояния или топологии в $\mathcal{F}$.

Полином f одной переменной степени d может иметь или не иметь корень в $\mathcal{F}$, то есть элемент $r \in \mathcal{F}$, такой, что $f(r) = 0$. В первом случае многочлен первой степени $X - r$ делит f; если $(X - r)^m$ есть высшая степень от $(X - r)$, которая делит f, говорят, что r – корень кратности m. Ввиду единственности факторизации общее

число корней многочлена f в $\mathcal{F}$ с учетом кратности не превышает d. Если многочлен $f \in \mathcal{F}$ имеет кратный корень r, то r будет и корнем производной f', и, следовательно, н.о.д. многочленов f и f'.

Пусть дан многочлен $f(X) \in \mathcal{F}[X]$ одной переменной и имеется расширение $\mathcal{K}$ поля $\mathcal{F}$ такое, что $f(X) \in \mathcal{K}[X]$ разлагается в произведение линейных множителей (или, что эквивалентно) имеет d корней в $\mathcal{K}$ с учетом кратностей, где d – степень многочлена $f(X)$, и такое, что $\mathcal{K}$ является наименьшим расширением поля, содержащим эти корни. Тогда $\mathcal{K}$ называется *полем разложения многочлена f над полем $\mathcal{F}$*. Такое поле единственно с точность до изоморфизма.

Пример 1.3. Поле $\mathcal{Q}(\sqrt{2})$ является полем разложения многочлена $f(X) = X^2 - 2 \in \mathcal{Q}[X]$. Для получения поля разложения многочлена $f(X) = X^3 - 2 \in \mathcal{Q}[X]$ надо присоединить к $\mathcal{Q}$ как $\sqrt[3]{2}$, так и $\sqrt{-3}$. (Напомним, что нетривиальными кубическими корнями 1 являются $(-1 \pm \sqrt{-3})/2$, так что присоединение $\sqrt{-3}$ эквивалентно присоединению всех кубических корней 1.)

Если поле $\mathcal{F}$ обладает тем свойством, что всякий многочлен с коэффициентами из $\mathcal{F}$ разлагается на линейные множители, то есть всякий такой многочлен имеет корни из $\mathcal{F}$, то оно называется алгебраически замкнутым. Наименьшее замкнутое поле, содержащее $\mathcal{F}$ называется алгебраическим замыканием поля $\mathcal{F}$ и обозначается $\bar{\mathcal{F}}$. Например, алгебраическим замыкание поля $\mathcal{R}$ действительных чисел является поле $\mathcal{C}$ комплексных чисел.

Векторное пространство над полем, алгебраические расширения поля

Векторное пространство над полем $\mathcal{F}$ определяется так же, как векторное пространство над полем $\mathcal{R}$ действительных чисел, то есть как множество векторов, любая линейная комбинация которых принадлежит полю, причем сложение дистрибутивно относительно умножения на скаляр. Векторное пространство имеет базис. Число элементов в базисе векторного пространства определяет его *размерность.*

Заметим, что векторное пространство размерности d над полем $\mathcal{F}$ можно получить из кольца многочленов $\mathcal{F}[X]$ подстановкой в каждый многочлен корня x любого неприводимого многочлена из

этого кольца степени d.

Действительно, любой элемент из $\mathcal{F}[x]$ может быть представлен как линейная комбинация

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_{d-1} x^{d-1} \tag{1.1}$$

степеней $1, x, x^2, \ldots, x^{d-1}$: возникающие после подстановки x в многочлен из кольца термы x^{d+i} должны заменяться на $x^{d+i} - x^i \times p(x)$, так как

$$x^d - x^i \times p(x) = x^i \times (x^d - p(x)) = x^i \times x^d = x^{d+i}, \tag{1.2}$$

поскольку x – корень многочлена $p(X)$ и $p(x) = 0$.

Таким образом, эти степени x образуют базис векторного пространства размерности d над полем $\mathcal{F}$.

Расширение поля $\mathcal{F}$ как бо̀льшее поле, содержащее $\mathcal{F}$, совпадает (как множество) с векторным пространством над полем $\mathcal{F}$. При этом операция сложения поля то же, что и операция сложения векторного пространства, а коммутативная операция умножения, такая, что каждый ненулевой элемент имеет обратный, может быть определена в общем случае несколькими способами, рассматриваемыми ниже. Мы называем такое поле *конечным* расширением, если оно является векторным пространством конечной размерности. При этом размерность векторного пространства определяет степень конечного расширения поля. Общим способом получения расширения поля является *присоединение* нового элемента x из некоторого расширения $\mathcal{H}$ поля $\mathcal{F}$ к этому полю. Будем обозначать получающееся при этом поле $\mathcal{K} = \mathcal{F}(x)$. От выбора этого элемента зависит операция умножения расширения поля, что будет показано далее.

Элемент x некоторого расширения $\mathcal{K}$ поля $\mathcal{F}$ называется алгебраическим над $\mathcal{F}$, если существует такой многочлен одной переменной $f(X) \in \mathcal{F}[X]$, что $f(x) = 0$. В этом случае имеется единственный нормированный неприводимый многочлен $p(X)$ в $\mathcal{F}[X]$, корнем которого является x, который называется *минимальным нормированным многочленом для* x. Любой другой многочлен, которому удовлетворяет x, делится на этот многочлен.

Всякий другой корень x' минимального многочлена от x называется сопряжением x над $\mathcal{F}$. Произведение всех сопряжений,

включая сам корень x, называется нормой. Если x' является сопряжением x, то поля $\mathcal{F}(x)$ и $\mathcal{F}(x')$ изоморфны.

Например, $\sqrt{2}$ имеет одно сопряжение над $\mathcal{Q}$, а именно $-\sqrt{2}$, и отображение $a + b\sqrt{2} \to a - b\sqrt{2}$ является автоморфизмом поля $\mathcal{Q}(\sqrt{2})$, образованного всеми действительными числами вида $a + b\sqrt{2}$, где a и b – рациональные числа.

Элемент g делит элемент f в поле $\mathcal{F}$ если существует элемент h, такой, что $f = g \cdot h$, $f, g, h \in \mathcal{F}$.

Если минимальный многочлен для x имеет степень d, то любой элемент из $\mathcal{F}(x)$ может быть представлен как линейная комбинация (1.1) степеней $1, x, x^2, \ldots, x^{d-1}$. Таким образом, эти степени образуют базис поля $\mathcal{F}(x)$ над $\mathcal{F}$, так что степень расширения, получающегося присоединением элемента x, такая же, как и степень минимального многочлена от x.

При таком выборе элемента x естественным образом определяется операция умножения получающегося расширения поля: результатом умножения двух элементов поля, представленных многочленами вида (1.1) является значение при $X = x$ результата умножения соответствующих многочленов кольца, то есть многочленов вида

$$a_0 + a_1 X + a_2 X^2 + \ldots + a_{d-1} X^{d-1}, \tag{1.3}$$

по правилам умножения в кольце $\mathcal{F}_p[X]$.

Это же значение является значением при $X = x$ остатка от деления результата умножения таких многочленов кольца на минимальный нормированный многочлен $p(X)$.

Действительно, поскольку x является корнем этого многочлена, то возникающие в процессе умножения и упрощения в соответствии с тождеством (1.2) представления и будут значением указанного остатка.

Ввиду изоморфизма представлений (1.1) и (1.3), сохраняющего результаты операции умножения по модулю неприводимого многочлена алгоритм операции умножения в расширении поля можно представить непосредственно с использованием обозначений вида (1.1).

При этом для удобства корень x, степени которого образуют базис поля будем обозначать x и выражения (1.1) записывать в

виде суммы

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_{d-1} x^{d-1} \tag{1.4}$$

степеней
$1, x, x^2, \ldots, x^{d-1}$, составляющих так называемый полиномиальный или стандартный базис поля $\mathcal{F}(x)$.

Итак, для умножения двух полиномов в стандартном базисе достаточно умножить их представления в виде (1.1) по правилам умножения в кольце многочленов и затем взять остаток от деления на многочлен $f(x)$ по правилам деления в кольце многочленов. (Заметим, что $f(x) = 0$, однако в данном случае деление на 0 допустимо, так как этот процесс отображает деление на многочлен $f(X)$ и последующую подстановку корня x в окончательный результат.) Ниже операция умножения в кольце многочленов $\mathcal{F}[X]$ обозначается $\times$, а операция умножения в поле $\mathcal{F}(x)$ обозначается $\cdot$.

Наибольшим общим делителем (н.о.д.) двух элементов g, f поля $\mathcal{F}(x)$ называется элемент поля, представляемый нормированным многочленом наибольшей степени, который делит каждый из этих элементов.

Очевидно, что он получается подстановкой x в н.о.д. многочленов, $g(X)$ и $f(X)$ кольца многочленов $\mathcal{F}[X]$.

Алгоритм Евклида

Наибольший общий делитель (н.о.д.) многочленов $f(X)$ и $g(X)$ в кольце многочленов $\mathcal{F}[X]$, в частности в кольце $\mathcal{F}_p[X]$, (или многочленов $f(x)$ и $g(x)$ в расширении $\mathcal{F}_p(x)$ поля $\mathcal{F}_p$) можно представить через эти многочлены следующим образом

$$(f(X), g(X)) = p(X) \times f(X) + q(X) \times g(X). \tag{1.5}$$

или

$$(f(x), g(x)) = p(x) \cdot f(x) + q(x) \cdot g(x). \tag{1.6}$$

Для нахождения н.о.д. полиномов в кольце полиномов можно использовать различные варианты алгоритма Евклида. При этом можно получить и представление н.о.д. в виде (1.5) или (1.6).

Последнее можно использовать для инвертирования данного многочлена $f(x)$ в поле $\mathcal{F}_(x)$ Действительно, если $g(X)$ неприводимый над полем $\mathcal{F}_p$ многочлен, корень x которого используется

для построения расширения $\mathcal{F}_p(x)$ этого поля, то

$$(f(x), g(x)) = p(x) \times f(x) + q(x) \times g(x) = 1.$$

Но $q(x) \times g(x) = 0$, и, следовательно,

$$p(x) \times f(x) = 1,$$

то есть, $p(x) = f^{-1}(x)$.

1.1.2 Некоторые свойства конечных полей

Пусть $\mathcal{F}_q$ – поле с конечным числом элементов q. Ясно, что конечное поле не может быть полем характеристики 0 и его характеристикой является число p. Тогда поле $\mathcal{F}_q$ содержит поле Z/pZ и является векторным пространством конечной размерности над $\mathcal{F}_p$. Пусть f обозначает его размерность как $\mathcal{F}_p$-векторного пространства. Выбрав базис, можно установить взаимно однозначное соответствие между элементами этого f-размерного векторного пространства и множеством всех f-кортежей элементов из $\mathcal{F}_p$. Отсюда следует, что должно быть $q = p^f$ элементов в $\mathcal{F}_q$. То есть q является *степенью характеристики* p.

Мы скоро покажем, что для каждой степени $q = p^f$ любого простого числа p существует единственное с точностью до изоморфизма поле из q элементов. Но сначала мы исследуем мультипликативный *порядок* ненулевых элементов из $\mathcal{F}_q$.

Существование мультипликативных образующих конечного поля

Имеется $q - 1$ ненулевых элементов и, по определению поля, они составляют абелеву группу по умножению. Это означает, что произведение двух любых таких элементов не равно нулю и выполняются свойства группы. Так по теореме Лагранжа порядок любого элемента $a \in \mathcal{F}_q^*$ делит $q - 1$.

Следующая теорема отражает основной факт о конечных полях. Она говорит о том, что ненулевые элементы любого конечного поля составляют *циклическую группу,* то есть являются степенями одног и того же элеиента.

Теорема 1.2. *Каждое конечное поле имеет образующий элемент. Если g является образующим элементом группы $\mathcal{F}_q^*$, то g^j*

также является образующим элементом тогда и только тогда, когда н.о.д. $(j, q-1)$ *элементов* j *и* $q-1$ *равен* 1. *Так общее число образующих группы* $\mathcal{F}_q^*$ *составляет* $\varphi(q-1)$, *где* φ – *функция Эйлера.*

Справедлива также следующая лемма, используемая при доказательстве этой теоремы

Лемма 1.1. *Для всякого целого числа* $N > 1$ *имеет место*

$$\Sigma_{d|n}\varphi(d) = N.$$

Следствие 1.1. *Для всякого простого числа существует целое* g, *такое, что степени* g *порождают все ненулевые классы по модулю* p.

Пример 1.4. Мы можем получить все остатки по модулю 19 от 1 до 18, используя подходящие степени 2; последовательные степени 2 после приведения по модулю 19 следующие: 2,4,8,16,13,7,14,9,18,17,15,11,3,6,12,5,10,1.

Существование и единственность конечных полей с числом элементов, равным степени простого числа

Как существование, так и единственность доказываются демонстрацией того, что конечное поле из $q = p^f$ элементов является полем разложения многочлена $X^q - X$.

Следующая теорема утверждает, что для каждой степени $q = p^f$ простого числа имеется одно и с точность до изоморфизма только одно конечное поле с q элементами.

Теорема 1.3. *Если* $\mathcal{F}_q$ *является полем из* $q = p^f$ *элементов, то каждый его элемент удовлетворяет уравнению* $X^q - X = 0$, *и это поле в точности состоит из корней этого уравнения. Обратно, для каждой степени* $q = p^f$ *простого числа поле разложения многочлена* $X^q - X$ *над* $\mathcal{F}$ *содержит* q *элементов.*

Следствие 1.2. *Если* $\mathcal{F}_q$ *является полем из* $q = p^f$ *элементов, то каждый его ненулевой элемент удовлетворяет уравнению* $X^{q-1} = 1$.

Это утверждение является аналогом малой теоремы Ферма.

Следствие 1.3. *Если* $\mathcal{F}_q$ *является полем из* $q = p^f$ *элементов, то для каждого его ненулевого элемента* a *обратным по умножению является элемент* a^{q-2}.

Отметим также фундаментальный факт о полях характеристики p.

Лемма 1.2. *В любом поле характеристики p выполняется тождество $(a+b)^p = a^p + b^p$.*

Таким образом, возведение в степень сохраняет операции поля. Этот факт обобщается следующей теоремой.

Теорема 1.4. *Пусть $\mathcal{F}_q$ – конечное поле из $q = p^f$ элементов и пусть σ есть отображение, сопоставляющее каждому элементу его p-ю степень: $\sigma(a) = a^p$. Тогда σ является автоморфизмом поля $\mathcal{F}_q$. Элементами, сохраняемыми при автоморфизме, являются элементы простого поля $\mathcal{F}_p$, и только они. При этом f-я степень этого отображения, но никакая меньшая степень, является тождественным отображением.*

Теорема 1.5. *Подполями поля $\mathcal{F}_{p^f}$ являются все поля $\mathcal{F}_{p^d}$ для d, делящих f. Если элемент поля $\mathcal{F}_{p^f}$ присоединяется к $\mathcal{F}_p$, получается одно из таких полей.*

Пример

До сих пор наши рассуждения имели теоретический характер. практически же рассматривались конечные поля вида Z/pZ. Теперь посмотрим, как работать с конечным расширением $\mathcal{F}_p$. Напомним, как в случае поля $\mathcal{Q}$ рациональных чисел мы работаем с таким расширением, как $\mathcal{Q}(\sqrt{2})$. Мы получаем это поле, используя корень x уравнения $X^2 - 2 = 0$ и рассматриваем всевозможные выражения вида $a + bx$, которые складываются и умножаются обычным способом, причем x^2 всегда заменяется на 2. В случае $\mathcal{Q}(\sqrt[3]{2})$ мы работаем с выражениями вида $a + bx + cx^2$, заменяя всегда x^3 на 2. Мы можем применять этот общий прием и при работе с конечными полями.

Пример 1.5. Для построения $\mathcal{F}_9$ возьмем любой квадратичный многочлен в $\mathcal{F}_3[X]$, не имеющий корней в $\mathcal{F}_3$. Перебирая всевозможные варианты таких многочленов и проверяя в каждом случае, не являются ли элементы $0, \pm 1 \in \mathcal{F}_3$ корнями соответствующих многочленов, убедимся, что имеется только три неприводимых квадратичных многочлена: X^2+1, $X^2 \pm X - 1$. Если, например, мы выберем в качестве x корень многочлена X^2+1 и и обозначим его как i, то элементами поля будут всевозможные комбинации

вида $a + bi$, где a и b принимают значения $0, \pm 1$. Как видим, это поле по виду похоже на арифметику Гауссовых целых (множество комплексных чисел с целыми действительной и мнимой составляющими) и отличается тем, что коэффициенты a и b берутся из $\mathcal{F}_3$.

Заметим, что присоединяемый элемент i не является образующим элементом группы $\mathcal{F}_9^*$, так как имеет порядок 4, меньший, чем $q-1=8$. если же мы присоединим корень x многочлена X^2-X-1, мы можем иметь все ненулевые элементы поля $\exists$, беря последовательные степени x. (Напомним, что x^2 заменяется на $x+1$, поскольку x удовлетворяет уравнению $X^2 = X + 1$) : $x^1 = x$, $x^2 = x+1$, $x^3 = -x+1$, $x^4 = -1$, $x^5 = -x$, $x^6 = -x-1$, $x^7 = x-1$, $x^8 = 1$.

О неприводимых многочленах

Неприводимые многочлены, все корни которых являются образующими элементами мультипликативной группы, называются *примитивными.* По теореме 1.1 имеется $\varphi(8) = 4$ образующих группы $\mathcal{F}_9^*$: два из них являются корнями многочлена $X^2 - X - 1$ и два – корнями многочлена $X^2 + X - 1$. (Второй корень многочлена $X^2 - X - 1$ является сопряжением для x,) а именно $\sigma(x) = x^3 = -x + 1$). Из оставшихся четырех ненулевых элементов два ($\pm i = \pm(x+1)$) являются корнями многочлена X^2+1 и другие два (± 1) – ненулевые элементы $\mathcal{F}_3$ (они – корни многочленов первой степени $X \pm 1$.)

Теорема 1.6. *Для каждого $q = p^f$ многочлен $X^q - X$ разлагается в $\mathcal{F}_p[X]$ в произведение всех неприводимых многочленов степени d, делящих f.*

Следствие 1.4. *Если f – простое число, то имеется $(p^f - p)/f$ различных нормированных неприводимых многочленов степени f в $\mathcal{F} - p[X]$.*

Заметим, что $(p^f - p)/\mathrm{a}$ – целое по малой теореме Ферма.

1.1.3 Упражнения

1.1. Для p=2,3,5,7,11,13 и 17 найти наименьшее положительное число, являющееся образующим элементом для группы $\mathcal{F}_p^*$ и определить, сколько чисел из $1, 2, 3, \ldots, p-1$ являются образующими.

1.2. Для каждого положительного $d \leq 6$ найти число неприводимых многочленов над $\mathcal{F}_2$ степени d и построить эти многочлены.

1.3. Для каждого положительного $d \leq 4$ найти число нормированных неприводимых многочленов над $\mathcal{F}_3$ степени d и построить эти многочлены.

1.4. Написать и программы для операции умножения в полях, порождаемых присоединением к $GF(2)$ неприводимого полинома, а также программы вычисления обратного элемента.

1.2 Поля Галуа $GF(2^n)$

1.2.1 Еще раз о полях Галуа

Как было отмечено в первом разделе, простые конечные поля $\mathcal{F}_p$, где p - простое число, и их расширения $\mathcal{F}_{p^n}$ называются также полями Галуа и обозначаются $GF(p)$ и $GF(p^n)$ соответственно. При этом алгебраическое расширение $GF(p^n)$ образуется присоединением к полю $GF(p)$ корня x некоторого неприводимого многочлена степени n над полем $GF(p)$, то есть $GF(p^n) = GF(p)(x)$. В данном разделе мы рассматриваем поля Галуа $GF(2^n) = GF(2)(x)$, возникающие в результате присоединения к простейшему полю Галуа $GF(2)$ корня x неприводимого многочлена степени n над этим простейшим полем.

1.2.2 Кольцо многочленов над $GF(2)$

Кольцо многочленов $GF(2)[X]$ одной переменной X над полем $GF(2)$ определяется как множество всех конечных сумм степеней переменной X. При этом аддитивной единицей является нулевой многочлен, а мультипликативной единицей – многочлен 1. Операция сложения является поразрядной операцией, при выполнении которой степени переменной, встречающиеся в обоих слагаемых исключаются, а степени, встречающиеся только в одном из слагаемых включаются как члены многочлена-произведения. То есть сложение заключается в покомпонентном (для каждой степени) сложении по модулю 2. Операция умножения соответствует обычному умножению многочленов с тем отличием, что все операции сложения и умножения над коэффициентами заменяются

соответствующими операциями поля $GF(2)$. Таким образом, операции кольца многочленов над $GF(2)$ можно описать следующим образом.

Операция *сложения* сопоставляет двум многочленам степени не выше, чем $n-1$

$$p_1(X) = \sum_{i=0}^{n-1} a_i X^i \text{ и } p_2(X) = \sum_{i=0}^{n-1} b_i X^i;$$

их сумму

$$p_1(X) + p_2(X) = \sum_{i=0}^{n-1} (a_i \oplus b_i) X^i.$$

Здесь и ниже в формулах, подобных формуле в правой части, имеются в виду логические операции сложения и умножения.

Результатом операции *умножения* многочленов $p_1(X) = \sum_{i=0}^{n-1} a_i X^i$ и $p_2(X) = \sum_{i=0}^{n-1} b_i X^i$ является многочлен

$$p(X) = p_1(X) \times p_2(X) = \sum_{i=0}^{2n-2} c_i X^i,$$

где $c_i = \sum_{t+l=i} a_t b_l$.

Кольцо многочленов не является полем, так как не для всякого многочлена имеется обратный по умножению многочлен.

В кольце многочленов можно определить операцию деления, сопоставляющую многочленам $f(X)$ и $g(X)$ многочлен $r(X)$ степени меньшей степени многочлена $g(X)$, такой, что существует многочлен $q(X)$, удовлетворяющий равенству $f(X) = q(X) \times g(X) + r(X)$.

Если при этом многочлен $r(X) = 0$, то говорят, что $g(X)$ делит $f(X)$, что записывают как $g|f$.

Наибольший общий делитель (н.о.д.) двух многочленов $f(X)$ и $g(X)$ можно представить через эти многочлены следующим образом

$$(f(X), g(X)) = p(X) \times f(X) + q(X) \times g(X).$$

Для нахождения н.о.д. полиномов можно использовать алгоритм Евклида.

1.2.3 Расширения поля $GF(2)$ – поля Галуа $GF(2^n)$

Пусть f(X) – неприводимый над $GF(2)$ многочлен степени n и x – его корень в некотором расширении поля $GF(2)$. Тогда все линейные комбинации вида

$$1 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}$$

образуют поле Галуа $GF(2^n)$ с операцией умножения, определяемой неприводимым многочленом $f(X)$.

Эта операция отличается от операции умножения многочленов над полем $GF(2)$, то есть от операции умножения в кольце многочленов тем, что возникающие при умножении одночлены x^{n+i} заменяются на $x^i \times (f(x)+x^n)$, так как $f(x)+x^n = x^n$, а x – корень $f(X)$. Эти замены можно производить в процессе выполнения умножения или же по окончании процесса умножения над полем $GF(2)$. Во всех случаях результатом замены будет остаток от деления (в кольце многочленов) промежуточного или окончательного результата операции умножения над $GF(2)$ на $f(x)$.

Операция сложения не отличается от операции сложения многочленов над полем $GF(2)$.

Простейшее поле Галуа $GF(2)$ соответствует в указанном смысле многочлену $X + 1$.

Пример 2.1. Многочлен $X^2 + X + 1$ порождает поле Галуа $GF(2^2)$ с операциями сложения $(+)$ и умножения $(\cdot)$, определяемыми таблицами (при умножении элементов как элементов кольца в каждом случае следует заменять X^2 на $X + 1$):

$+$	00	10	01	11
00	00	00	00	00
10	10	00	11	01
01	01	11	11	10
11	00	01	10	00

$\cdot$	00	10	01	11
00	00	00	00	00
10	00	10	01	11
01	00	01	11	10
11	00	11	10	01

1.2.4 Алгоритм Евклида

Алгоритм Евклида вычисления н.о.д. двух многочленов такой же, как и для чисел. Классический вариант алгоритма Евклида для чисел следующий

Даны числа r_{-2} и r_{-1},
найти (r_{-2}, r_{-1}) – их наибольший общий делитель.
Положить $k = 0$,
Пока $r_n \neq 0$ вычислять, сохраняя a_k,
$a_k = [r_{k-2}/r_{k-1}], r_k = -a_k r_{k-1} + r_{k-2}$,
$n = k,\ GCD = r_{k-1},\ k = k + 1.$

Принять $k = n - 1$,
Положить $p_k = 0,\ q_k = 1.$
Пока $k \neq -2$ вычислять
$p_{k-1} = q_k,\ q_{k-1} = p_k - a_k q_k,\ k = k - 1$

Получаемые по этому алгоритму числа $GCD = r_{n-1},\ p_{-1}$ и q_{-1} таковы, что

$$p_{-1} r_{-2} + q_{-1} r_{-1} = r_{n-1} = (r_{-1}, r_{-1}).$$

Действительно, легко проверить, что для всех k выполняется

$$r_{n-1} = q_k r_k + p_k r_{k-1} :$$

так как $r_k = r_{k-2} - a_k r_{k-1}$, то $r_k = r_{k-2} - a_k r_{k-1}$ и, следовательно,

$$q_k r_k + p_k r_{k-1} = q_k (r_{k-2} - a_k r_{k-1}) + p_k r_{k-1} =$$

$$= (p_k - a_k q_k) r_{k-1} + q_k r_{k-2}.$$

Предлагается сформулировать и доказать аналогичный алгоритм для элементов поля $\mathcal{F}(x)$ – многочленов $r^{(-2)}, r^{(-1)}$, где

$$r^{(-2)}(x) = \sum_1 r_i^{(-2)} x^i,\ r^{(-1)}(x) = \sum_i r_i^{(-1)} x_i.$$

Решение.

Даны многочлены $r^{(-2)}$ и $r^{(-1)}$,
найти $(r^{(-2)}, r^{(-1)})$ – их наибольший общий делитель.
Положить $k = 0$,
Пока $r^n \neq 0$вычислять, сохраняя $a^{(k)}$,
$a^{(k)} = [r^{(k-2)}/r^{(k-1)}], r^{(k)} = -a^{(k)} r^{(k-1)} + r^{(k-2)}$,
$n = k,\ GCD = r^{(k-1)},\ k = k + 1.$

Принять $k = n - 1$,
Положить$p^{(k)} = 0,\ q^{(k)} = 1$.Пока $k \neq -2$вычислять
$p^{(k-1)} = q^{(k)},\ q^{(k-1)} = p^{(k-1)} - a^{(k)} q^{(k)},\ k = k - 1.$

Здесь $a^{(k)} = [r^{(k-2)}/r^{(k-1)}]$ – частное от деления многочлена r_{k-2} на многочлен r_{k-1}.

Получаемые по этому алгоритму многочлены $GCD = r^{n-1}$, $p^{(-1)}$ и $q^{(-1)}$ таковы, что

$$p^{(-1)} r^{(-2)} + q^{(-1)} r^{(-1)} = r^{(n-1)} = (r^{(-1)}, r^{(-1)}). \quad (2.1)$$

Приведем пример работы этого алгоритма с многочленами над $\mathcal{F}_2$.

Пример 2.2. Пусть $f(x) = r^{(-2)}(x) = x^4 + x^3 + x^2 + 1$, $g(x) r_{(-1)}(x) = x^3 + 1$. найдем н.о.д. и представим его в виде

$p(x) \times f(x) + q(x) \times g(x) \quad (2.2)$

С помощью операции деления многочленов получаем последовательность равенств, позволяющую заключить, что н.о.д.$(f, g) = x + 1$.

Мы имеем:

$$r^{(-2)}(x) = (x + 1) \times r^{(-1)}(x) + (x^2 + x),$$

т.е. $a^{(0)}(x) = x + 1,\ r^{(0)}(x) = x^2 + x$

$$r^{(-1)}(x) = (x + 1) \times (x^2 + x) + (x + 1),$$

т.е. $a^{(1)}(x) = x + 1,\ r^{(1)}(x) = x + 1,$

$$x^2 + x = x \times (x + 1).$$

т.е. $a^{(2)}(x) = x,\ r^{(2)}(x) = 0,\ n = 2.$

$(r^{(-2)}, r^{(-1)}) = r^{(n-1)} = r^{(1)} = x + 1$.

Полагая $p^{(n-1)} = p^{(1)} = 0,\ q^{(n-1)} = p^{(1)} - 1$ и просматривая эту колонку в обратном порядке, получим полиномы $p^{(-1)}$ и $q^{(-1)}$ и затем выражение (2.4) для н.о.д. через полиномы $f(x)$ и $g(x)$:

$$p^{(0)}(x) = q^{(1)}(x) = 1,\ q^{(0)}(x) = p^{(1)}(x) + a^{(1)}(x)q^{(1)}(x) = x + 1,$$

$$p^{(-1)}(x) = q^{(0)}(x) = x + 1,\ q^{(-1)}(x) = p^{(0)}(x) + a^{(0)}(x)q^{(0)}(x) = x^2,$$

$$(r^{(-2)}(x), r^{(-1)}(x)) = (x + 1)r^{(-2)}(x) + x^2 r^{(-1)}(x).$$

Отметим, что использованный нами классический вариант алгоритма Евклида обладает тем недостатком, что для получения представления н.о.д. через исходные многочлены в виде (2.2) необходимо сохранить все промежуточные результаты вычисления н.о.д.. Этого недостатка лишен рассматриваемый ниже вариант этого алгоритма.

1.2.5 Алгоритм Евклида (вариант цепных дробей) для чисел

Даны числа r_{-2} и $r_{(-1)}$,
найти (r_{-2}, r_{-1}) – их наибольший общий делитель.
Положить
$p_{-2} = 0,\ p_{-1} = 1, q_{-2} = 1,\ q_{-1} = 0,\ k = 0$
и пока $r_n \neq 0$ вычислять
$r_k = -a_k r_{k-1} + r_{k-2}, 0 \leq r_k < r_{k-1},$
$p_k = a_k p_{k-1} + p_{k-2},$
$q_k = a_k q_{k-1} + q_{k-2}.$
$n = k,\ GCD = r_{n-1},\ k = k + 1.$

Получаемые при этом числа $r_{n-1},\ p_{n-1}$ и q_{n-1} удовлетворяют уравнению (2.4).

Действительно, легко проверить, что

$$q_k r_{k+1} + q_{k+1} r_k = q_{k-1} r_k + q_k r_{k-1},$$

$$p_k r_{k+1} + p_{k+1} r_k = p_{k-1} r_k + p_k r_{k-1}.$$

$$q_{k+1} p_k - p_{k+1} q_k = -(q_k p_{k_1} - p_k q_{k-1}).$$

Начиная с $k = -1$ и проводя вычисления по индукции, получим, что для всех $k \geq -1$

$$q_{k-1}r_k + q_k r_{k-1} = r_{-1},$$

$$p_{k-1}r_k + p_k r_{k-1} = r_{-2},$$

$$q_k p_{k-1} - p_k q_{k-1} = (-1)^k.$$

При $k = n$ с учетом $r_n = 0$ получаем

$$q_n r_{n-1} = r_{-1},$$

$$p_n r_{n-1} = r_{-2},$$

$$r_{n-1}(q_n p_{n-1} - p_n q_{n-1}) = (-1)^n r_{n-1},$$

откуда имеем

$$r_{-1}p_{n-1} - r_{-2}q_{n-1} = (-1)^n r_{n-1} = (-1)^n (r_{-2}, r_{-1}).$$

Предлагается сформулировать и доказать аналогичные алгоритмы для элементов поля $\mathcal{F}(x)$ – многочленов $r^{(-2)}, r^{(-1)}$, где

$$r^{(-2)}(x) = \sum_1 r_i^{(-2)} x^i, \;\; r^{(-1)}(x) = \sum_i r_i^{(-1)} x_i.$$

Решение.

Полагаем
$p^{(-2)} = 0, \;\; p^{(-1)} = 1,$
$q^{(-2)} = 1, \;\; q^{(-1)} = 0$
и для $k = 0, 1, \ldots$ вычисляем
$p^{(k)} = a^{(k)} p^{(k-1)} + p^{(k-2)},$
$q^{(k)} = a^{(k)} q^{(k-1)} + q^{(k-2)}.$
Так же как и для чисел получаем
$r^{(n)} = 0,$
$p^{(n)} r^{(n-1)} = r^{(-2)},$
$q^{(n)} r^{(n-1)} = r^{(-1)},$
$r^{(-1)} p^{(n-1)} - r^{(-2)} q^{(n-1)} = (-1)^n r^{(r-1)}.$

Так как $\deg a^{(n)} > 0$, то, очевидно,

$$\deg p^{(n-1)} < \deg \frac{r^{(-2)}}{r^{(n-1)}} \text{ и } \deg q^{(n-1)} < \deg \frac{r^{(-1)}}{r^{(n-1)}}.$$

Как видно из этих выкладок, любой общий делитель многочленов $r^{(-2)}$ и $r^{(-1)}$ должен также делить $r^{(-1)}p^{(n-1)}$ и $r^{(-2)}q^{(n-1)}$, и, следовательно, любой общий делитель $r^{(-2)}$ и $r^{(-1)}$ делит $r^{(n-1)}$. Это доказывает, что $r^{(r-1)}$ — наибольший общий делитель этих полиномов. Отсюда следует, что наибольшие общие делители этих многочленов должны делить друг друга. В случае чисел это влечет единственность н.о.д., а в случае многочленов указывает на то, что н.о.д. двух многочленов определяется с точностью до скалярных множителей — элементов поля F. Неоднозначность можно устранить нормированием, то есть путем использования в качестве н.о.д. полинома со старшим коэффициентом 1.

Как видим, неприводимый полином не имеет делителей, кроме скаляров и скалярных кратных самого себя.

Примечание. Рассмотренный вариант алгоритма Евклида называется вариантом цепных дробей, так как частное p_n/q_n, представляют собой цепную дробь

$$p_n/q_n = r_2/r_1 = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\cdots + \cfrac{1}{a_n}}}}},$$

а частные p_k/q_k являются соответствующими её начальными частями:

$$p_k/q_k = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\cdots + \cfrac{1}{a_k}}}}}.$$

В данном варианте алгоритма на каждом шаге необходимо запоминать не более семи чисел или полиномов.

Пример 2.3. (Стандартный подход, вычисление н.о.д и коэффициентов $a^{(k)}$) и затем $p^{(k)}, q^{(k)}$.

Пусть (коэффициенты из бинарного поля $\{0, 1\}$)

$$r^{(-2)} = x^5 + x^2 + 1,\ r^{(-1)} = x^4 + x + 1.$$

Вычисления $a^{(k)}$ и $r^{(k)}$ (по $r^{(k-2)}, r^{(k-1)}$) представлены в таблице

k	$r^{(k-2)} =$	$a^{(k)}$	$r^{(k-1)}$	$+r^{(k)}$
0	$x^5 + x^2 + 1 =$	x	$x^4 + x + 1$	$+(x + 1)$
1	$x^4 + x + 1 =$	$x^3 + x^2 + x$	$x + 1$	$+1$
2	$x + 1 =$	$x + 1$	1	$+0$

Получилось:

$$r^{-2} = x^5 + x^2 + 1,\ r^{(-1)} = x^4 + x + 1,\ r^{(0)} = x + 1,\ a^{(0)} = x,$$

$$r^{(1)} = 1,\ a(1) = x^3 + x^2 + x,\ r^{(2)} = 0,\ a^{(2)} = x + 1.$$

Многочлены $p^{(k)}$ с использованием сохраненных коэффициентов $a^{(k)}$ ($q^{(k)}$ вычисляются аналогично, но с другими начальными значениями)

k	$p^k =$	$a^{(k)}$	$p^{(k-1)}$	$+p^{(k-2)}$
-2	0			
-1	1			
0	$x =$	x	1	$+0$
1	$x^4 + x^3 + x^2 + 1 =$	$x^3 + x^2 + x$	x	$+1$
2	$x^5 + x^2 + 1 =$	$x + 1$	$x^4 + x^3 + x^2 + 1$	$+x$

Получилось:

$$p^{(-2)} = 0,\ p^{(-1)} = 1,\ p^{(0)} = x,$$

$$p^{(1)} = x^4 + x^3 + x^2 + 1,\ p^{(2)} = x^5 + x^2 + 1.$$

Пример 2.4. (Вариант цепных дробей, совмещенное вычисление $r^{(k)}$ и $p^{(k)}$, вычисление $q^{(k)}$ также может быть совмещено).

k	$r^{(k-2)} =$	$a^{(k)}$	$r^{(k-1)}$	$+r^{(k)}$	$p^k = a(k)\cdot p(k-1)+ +p^{(k-2)}$
-2					0
-1					1
0	$x^5 + x^2 + 1 =$	x	$x^4 + x + 1$	$+(x + 1)$	x
1	$x^4 + x + 1 =$	$x^3 + x^2 + x$	$x + 1$	$+1$	$x^4 + x^3 +x^2 + x$
2	$x + 1 =$	$x + 1$	1	$+0$	$x^5+ +x^2 + 1$

Упражнение. Используя алгоритм Евклида показать, что если неприводимый многочлен f делит произведение многочленов $g^{(1)}$ и $g^{(2)}$, то f делит либо $g^{(1)}$ либо $g^{(2)}$.

Решение. Пусть f не делит $g^{(1)}$, тогда $(f, g^{(1)}) = 1$ и $pf + qg^{(1)} = 1$. Значит, $pfg^{(2)} + qg^{(1)}g^{(2)} = g^{(2)}$. Поскольку f делит оба члена левой части равенства, он делит и правую его часть.

Обобщая, по индукции можно получить, что если неприводимый многочлен f делит произведение нескольких многочленов, то он делит хотя бы один из сомножителей. Отсюда, если $f^{(i)}$ и $g^{(k)}$ неприводимы и $\Pi_i f^{(i)} = \Pi_k g^{(k)}$, то каждый $f^{(i)}$ делит некоторый $f^{(k)}$, а каждый $f^{(k)}$ делит некоторый $f^{(i)}$. Это доказывает следующую теорему:

Теорема 2.1. *Любой нормированный многочлен над полем F однозначно записывается в виде произведения нормированных неприводимых многочленов над F.*

1.2.6 Мультипликативное обращение

Как найти полином $p(x)$ степени $< n$, такой, что $r(x)p(x) = 1 \bmod M(x)$, где $M(x)$ — неприводимый (неразложимый на множители) полином степени n. Это эквивалентно равенству $r(x)p(x) + M(x)q(x) = 1$ для некоторого полинома $q(x)$? Поскольку $M(x)$ неприводим $(r(x), M(x)) = 1$, то можно применить алгоритм нахождения н.о.д.

Начиная с $r^{(-2)} = M,\ r^{(-1)} = r,$
$p^{(-2)} = 0,\ p^{(-1)} = 1,\ q^{(-2)} = 1,\ q^{(-1)} = 0,$
используем алгоритм определения $a^{(k)}$ и $r^{(k)}$, таких, что
$r^{(k-2)} = a^{(k)} r^{(k-1)} + r^{(k)},\ \deg r^{(k)} < \deg r^{(k-1)}.$
Затем полагаем
$q^{(k)} = a^{(k)} q^{(k-1)} + q^{(k-2)}, p^{(k)} = a^{(k)} p_{(k-1)} + p^{(k-2)}.$
Итерацию продолжаем, пока не получится $r^{(n)} = 0.$
Решением является $p^{(n-1)}$.

Пример 2.5. Пусть $r(x) = x^4 + x + 1,\ M(x) = x^5 + x^2 + 1.$

Примем $r^{(-2)} = M(x),\ r^{(-1)} = r(x)$. Вычисление $p(x)$ представлено выше: $(p(x) = p^{(1)} = x^4 + x^3 + x^2 + 1$ из последней таблицы. Аналогично можно вычислить $q(x) = q^{(1)} = x^3 + x^2 + 1$. Очевидно, что

$$r(x)p(x) = x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + x + 1$$

$$M(x)q(x) = x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + x,$$

$$r(x)p(x) + M(x)q(x) = 1,$$

таким образом, $r(x)p(x) = 1$ **mod** $M(x)$.

1.3 Тесты неприводимости

В данном разделе изучаются способы проверки свойства неприводимости многочленов.

1.3.1 Тест на неприводимость. Алгоритм Берлекэмпа

Алгоритм Берлекэмпа определяющий, является ли данный многочлен неприводимым. Пусть нужно проверить многочлен $P(x)$ на неприводимость. Сначала вычислим производную $P'(x)$ многочлена $P(x)$.

Примечание. Производная $P'(x)$ образуется по правилу формального дифференцирования многочлена $P(x)$: если $P(x) = \Sigma_{i=0}^{n-1} a_i x^i$, то $P'(x) = \Sigma_{i=1}^{n-1} a_i x^{i-1}$.

Если наибольший общий делитель многочленов $P(x)$ и $P'(x)$ не равен 1, то многочлен $P(x)$ не является неприводимым, т.к. его разложение на неприводимые содержит квадрат многочлена (степени большей 0). В противном случае, через $P_i(x)$ обозначим остаток от деления многочлена x^{2i} на $P(x)$, $i = 1, 2, ..., \deg(P(x))-1$. Для сокращения времени вычисления этих остатков они определяются последовательно. Так, если вычислен $P_i(x)$, то $P_{i+1}(x)$ равен остатку от деления $x^2 P_i(x)$ на $P(x)$. Поэтому сначала вычисляем многочлен $R(x)$, равный остатку от деления $xP_i(x)$ на $P(x)$, а уже затем $P_{i+1}(x)$. Нетрудно видеть, что что $R(x) = xP_i(x)$, если $\deg(P_i(x)) + 1 < \deg(P(x))$ и $R(x) = xP_i(x) + P(x)$, в противном случае. Многочлен $P_{i+1}(x)$ равен остатку от деления $xR(x)$ на $P(x)$. Положим $R_i(x) = P_i(x) + x + i$, $i = 1, 2, ..., \deg(P(x)) - 1$. Далее вычислим ранг матрицы $A = (R_1(x), R_2(x), ..., R_{\deg(P(x))-1}(x))$. Если он оказался равным $\deg(P(x)) - 1$, то многочлен $P(x)$ неприводим, в противном случае этот многочлен приводим.

Пример 3.1. 1. Пусть нужно проверить на неприводимость многочлен $P(x) = x^4 + x + 1$. Имеем $P'(x) = 1$ и наибольший общий делитель многочленов $P(x)$ и $P'(x)$ равен 1. Поэтому разложение $P(x)$ в произведение неприводимых содержит только сомножители первой степени. Далее вычислим остатки $P_1(x), P_2(x), P_3(x)$ от деления многочленов x^2, x^4, x^6, соответственно на $P(x)$. Нетрудно видеть. что $P_1(x) = x^2$, $P_2(x) = x^2 + 1$, $P_3(x) = x^2 P_2(x) (\mathrm{mod} P(x)) = x^3 + x^2$. Поэтому $R_1(x) = x^2 + x$, $R2(x) = x^2 + 1 + x$, $R3(x) =$

$x^3 + x^2 + x^3 = x^2$. Таким образом необходимо вычислить ранг матрицы $A = (R1(x), R2(x), R2(x)) = (x^2+x, x^2+1+x, x^2)$. Третья компонента вектора A имеет наибольшую степень (она была выбрана потому, что имеет наименьшее число слагаемых среди многочленов с наибольшей степенью). Поэтому после первой итерации алгоритма приведения матрицы A к треугольному виду получаем $A = (x, 1 + x)$, $B = (x^2)$. Теперь в матрице A выберем первую строку и проделаем вторую итерацию этого алгоритма. Тогда получим $A = (1)$, $B = (x^2, x)$. После третьей итерации алгоритма приведения к треугольному виду матрица A не содержит элементов, а для матрицы B имеем: $B = (x^2, x, 1)$. Таким образом после окончания работы рассматриваемого алгоритма вектор B содержит три компоненты, т.е. исходный многочлен неприводим.

1.3.2 Нахождение неприводимых ¡малочленов¿

Для нахождения неприводимого малочленного полинома q заданной степени n по заданному простому модулю предлагается следующий алгоритм.

Выбирается случайный ¡малочлен¿ q. Проверяется, не является ли он приводимым. Если да, то выбираем следующий ¡малочлен¿ q. Проверка на неприводимость осуществляется следуюим образом. Вначале проверяем многочлен на наличие кратных корней. Для этого вычисляем его производную и находим наибольший общий делитель многочлена и его производной. Если он не равен 1, то очевидно, что q приводим, и опять выбираем следующий ¡малочлен¿. В противном случае продолжаем проверку на неприводимость. Для этого строится последовательность полиномов $q_{k+1} = q_k^p \bmod q$ начиная с полинома $q_0 = x$. Очевидно, что $q_k = x^{p^k} \bmod q$. Полином q будет неприводимым тогда и только тогда, когда $q_n = x \bmod q$ и для любого простого делителя s числа n наибольший общий делитель многочленов $q_{n/s} - x$ и q будет равен 1. Очевидно, что если для некоторого s имеем $q_{n/s} = x$, то многочлен q приводим, и вычислять последовательность q_k далее n/s не нужно.

Для оценки сложности алгоритма заметим, что возведение в степень p проводится по формуле $f(x)^p = f(x^p) \bmod q$ и имеет

сложность $O(pkn)$, где k — число одночленов в многочлене q, то есть в наших условиях эта сложность линейна. Поэтому сложность вычисления последовательности $q_k, k = 0, \ldots, n$ в худшем случае квадратична.

В приложении приведены фрагменты таблиц трех- и пятичленов, вычисленных с помощью авторской программы из [3].

Так например, в приложении А.2 приведен для иллюстрации полный список (с точностью до возвратных) всех неприводимых трехчленов для степеней n, где $2000 < n \leq 2100$, (в продолжение как бы того списка, который встречался авторам в доступной западной литературе и который был только для $n \leq 2000$). При предоставлении места авторы готовы опубликовать полные таблицы неприводимых трехчленов и пятичленов высоких степеней, в том числе и таких, какие не встречаются в доступной литературе (например, полная таблица пятичленов степени $n = 163$, фрагмент которой приведен в приложении В.1, состоит из нескольких десятков тысяч пятичленов). Отметим в заключение этого раздела, что выбранные в прилоденияx значения степени n используются при имплементации криптографических протоколов на основе эллиптической криптографии.

Глава 2

Имплементация операций в $GF(2^n)$

2.1 Классический алгоритм умножения над $GF(2)$

2.1.1 Постановка задачи

Рассматриваемые в данном разделе операции сложения и умножения многочленов над $GF(2)$ определяются следующим образом.

Операция *сложения* и *умножения* многочленов $p_1(x) = \sum_{i=0}^{n-1} a_i x^i$ и $p_2(x) = \sum_{i=0}^{n-1} b_i x^i$ определяются формулами (2.1) и (2.2) из предыдущей лекции.

Рассматриваются классический метод умножения и его модификации.

При реализации соответствующих алгоритмов применяются программные эвристики, учитывающие, что возможности одновременного выполнения однотипных поразрядных операций ограничиваются пределами машинного слова, но в то же время достаточно большой доступный объем оперативной памяти позволяет расширить параллелизм за счет табличной реализации групповых операций.

Если многочлен $f(x) = \sum_{i=0}^{n-1} a_i x^i$ представлен последовательностью коэффициентов

$$a_0, a_1, \ldots, a_n,$$

то его можно представить последовательностью

$$A^{(0)}, A^{(1)}, \ldots, A^{(k-1)}$$

из $k =]\frac{n}{s}[$ s-разрядных машинных слов

$$A^{(0)} = a_0, a_1, \ldots, a_{s-1},$$

$$A^{(1)} = a_s, a_{s+1}, \ldots, a_{2s-1},$$

$$\ldots$$

$$A^{(k-1)} = a_{(k-1)s}, a_{(k-1)s+1}, \ldots, a_{ns-1}.$$

В случае, когда n не делится на s, старшие $ks - n$ бит слова $A^{(k-1)}$ дополняются нулями.

В данной и следующей лекциях векторы коэффициентов, размещаемые в одном машинном слове, называются ¡длинными¿ целыми числами (им соответствует, например, тип данных long). Машинные слова могут разбиваться на более мелкие составные части, например, 32-разрядное ¡длинное¿ целое число A представляется четырьмя байтами

$$A[0], A[1], A[2], A[3].$$

Использование данных одновременно и как машинных слов и как байт обеспечивается, например, типом данных "union," а представление многочленов как последовательностей машинных данных осуществляется в виде классов.

Эти особенности предопределяют структуру декомпозиции операции как функции над ¡длинными¿ операндами, сводящей ее к операциям над машинными словами, на которые разбиваются эти операнды. В свою очередь, операции над машинными словами могут разлагаться в последовательность операций над их частями, допускающих, как правило, табличную реализацию.

Простейшим случаем такой декомпозиции является разложение операции сложения многочленов степени n на $k =]\frac{n}{s}[$ операций поразрядного сложения содержимого машинных слов длины s бит.

Декомпозиция умножения требует более тонкого анализа. Умножение над $GF(2)$ только в простейших случаях сводится к распараллеливанию стандартного метода умножения ¡столбиком¿. Один из возможных вариантов такого рода оказывается эффективным при достаточно малом количестве ненулевых коэффициентов одного из сомножителей.

Целью данной лекции является, таким образом, изучение эффективных имплементаций операции умножения многочленов над $GF(2)$, классическим методом с выбором модификации метода в зависимости от сложности операндов.

2.1.2 Элементарные многочлены. Таблица умножения

Будем рассматривать множество $E_2[x]$ многочленов переменной x с коэффициентами из поля E_2 из двух элементов.

Степень многочлена $P(x)$ обозначается $\deg P(x)$. Многочлены степени не выше $k-1$ будем называть *элементарными.* Число k является параметром и используется в дальнейшем для оптимизации времени работы алгоритмов. Возможные значения для k: 1,2,4,8,16,32.

Тогда многочлен $P(x)$ задается суммой

$$\sum_{i=0}^{s} Q_i(x)x^{ik},$$

где $Q_i(x)$ - элементарные многочлены. Наибольшее i, что $Q_i(x) \neq 0$, обозначим $\deg_k P(x)$.

Для разных алгоритмов предполагается использовать вообще говоря различные значения параметра k. Так для вычисления суммы $P_1(x)+P_2(x)$ многочленов $P_1(x)$ и $P_2(x)$ разумно положить $k=32$. Тогда элементарный многочлен определяется в памяти ПК машинным словом, а операция сложения многочленов сводится к логическому суммированию соответствующих элементарных многочленов.

Произведение двух многочленов P_1 P_2 вычисляется путем нахождения элементарных многочленов $Z_i(x)$, являющихся коэффициентами при x^{ik}, $1=0,1\ldots,\deg_k P_1 x+\deg_k P_2(x)$.

Перемножая два элементарных многочлена, получаем многочлен степени не выше $2k-2$, который задается суммой

$$Q_1(x)+Q_2(x)x^k,$$

где $Q_i(x)$ - элементарный многочлен, $i=1,2$.

Таким образом, произведение элементарных многочленов $P_1(x)P_2(x)$ представляется парой элементарных многочленов и может быть вычислено, например, по методу Берлекемпа-Петерсона: умножать элементарные многочлены $P_1(x)$ и $P_2(x)$,

$$P_2(x)=\sum_{i=0}^{k-1} a_i x^i,$$

предлагается путем вычисления произведения $P_1(x)x^i$ для каждого i, $i = 0, 1, \ldots, k-1$, для которого $a_i \neq 0$, и их суммирования.

Элементарные многочлены $Q_1(x)$ и $Q_2(x)$ такие, что

$$P_1(x)x^i = Q_1(x) + Q_2(x)x^{k-1}$$

определяются формулами

$$Q_1(x) = P_1(x)x^i(\mathrm{mod}\, x^k),$$

$$Q_2(x) = (P_1(x)(\mathrm{mod}\, x^{k-i}) + P_1(x)).$$

Пример. Пусть $P_1(x) = 1 + x + x^2,\ k = 3,\ i = 2.$

$$P_1(x)x^i = (1 + x + x^2)x^2 = x^2 + x^3 + x^4),$$

где

$$Q_1(x) = P_1(x)x^2(\mathrm{mod}\ x^3) = x^2,$$

$$Q_2(x) = P_1(x)(\mathrm{mod}\ x^{3-2}) + P_1(x) = x + x^2.$$

Рассмотрим алгоритм умножения элементарного многочлена $P_(x)$ на элементарный многочлен $P_2(x)$.

Даны векторы U и V длины k коэффициентов элементарных многочленов $P_1(x)$ и $P_2(x)$ в порядке возрастания степеней соответствующих термов многочленов, для формирования результата используется вектор Z длины $2k$ он образуется двумя векторами Z_1 и Z_2 длины k (Z_1 соответствует младшим, а Z_2 – старшим разрядам вектора Z)
Требуется вычислить произведение $Z = U \cdot V$.

1. $Z = 0$,
2. Выполнить k раз
Если$]V = 1$, то $Z_2 = Z_2 + U$,
$Z = Z[\leftarrow], V = V[\leftarrow]$.

$]V$ означает младший разряд вектора V, $[\leftarrow]$ – операция сдвига в сторону младших разрядов. Элементы векторов Z_1 и Z_2 определяют коэффициенты элементарных многочленов $Q_1(x)$ и $Q_2(x)$.

При небольших значения k (например, $k \leq 8$) операция умножения элементарных многочленов может выполняться с помощью

заранее составленной таблицы T_M умножения. Её строки и столбцы соответствуют различным элементарным многочленам, а элементы – произведениям элементарных многочленов (наборы бинарных коэффициентов записываются в порядке возрастания степеней соответствующих термов), что позволяет отказаться от поразрядных операций и работать с байтами как с минимальными неделимыми блоками данных.

Пример. Таблица умножения при $k = 2$ имеет вид:

	00	10	01	11
00	0000	0000	0000	0000
10	0000	1000	0100	1100
01	0000	0100	0010	0110
11	0000	1100	0110	1010

Примечание. При $k = 8$ таблица умножения занимает $2^8 \times 2^8 \times 2^1 = 2^{17}$ байт, или $2^7 = 128$ Кбайт.

2.1.3 Умножение многочленов с использованием таблицы умножения

Пусть

$$P_1(x) = \sum_{i=0}^{s} U_i(x)x^{ik}, \;\; P_2(x) = \sum_{j=0}^{p} V_j(x)x^{jk}.$$

$$P(x)\sum_{r=0}^{s+p} Z_r(x)x^{rk} = P_1(x) \times P_2(x).$$

Рассмотрим алгоритм вычисления "коэффициентов" $Z_r(x)$ произведения $P(x)$ многочленов $P_1(x)$ и $P_2(x)$.

Пусть $\deg_k P_1(x) = s$ и $\deg_k P_2 = p$. "Коэффициенты" $U_i(x)$ и $V_j(x)$ будем записывать в виде бинарных векторов длины k, образующих элементы массивов $U[s+1]$ и $V[p+1]$, "коэффициенты" $Z_r(x)$ произведения $P(x)$ будем вычислять как элементы массива $Z[s+p+1]$. Будем обозначать $\{U[i]\times V[j]\}_1$ k младших разрядов вектора, представляющего произведение элементарных многочленов U_i и V_j, а $\{U[i] \times V[j]\}_2$ – k старших разрядов этого произведения (эти векторы можно брать непосредственно из заранее составленной таблицы умножения). Алгоритм умножения можно описать

теперь следующим образом

$1. Z = 0;$
2. Для i $= 0,$ s
Для j $= 0,$ p
$Z_{i+j} = Z_{i+j} + \{U[i] \times V[j]\}_1;$
$Z_{i+j+1} = Z_{i+j+1} + \{U[i] \times V[j]\}_2$

Пример. Пусть $k = 8$ и нужно перемножить многочлены

$$P_1(x) = (1 + x^2)x^8 + (x^3 + 1) \text{ и } P_2(x) = (1 + x + x^7)x^{16} + x^6.$$

Тогда

$$P_1(x)P_2(x) =$$

$$= (1+x^2)(1+x+x^7)x^{24}+(x^3+1)(1+x+x^7)x^{16}+(1+x^2)x^6x^8+(x^3+1)x^6.$$

Обращаясь к заранее вычисленной таблице произведений элементарных многочленов, получаем

$$(1 + x^2)(1 + x + x^7) = xx^8 + (1 + x + x^2 + x^3 + x^7),$$

$$(x^3 + 1)(1 + x + x^7) = x^2x^8 + (1 + x + x^3 + x^4 + x^7),$$

$$(1 + x^2)x^6 = x^8 + x^6, \ (x^3 + 1)x^6 = xx^8 + x^6.$$

Откуда

$$P_1(x)P_2(x) = xx^{32} + (1 + x + x^2 + x^3 + x^7)x^{24} + x^2x^{24} +$$

$$+(1 + x + x^3 + x^4 + x^7)x^{16} + x^{16} + x^6x^8 + xx^8 + x^6 =$$

$$= xx^{32} + (1 + x + x^3 + x^7)x^{24} + (x + x^3 + x^4 + x^7)x^{16} + (x + x^6)x^8 + x^6.$$

Представим эти вычисления в виде таблицы.

Примечания. 1. В круглых скобках $(1), \ldots, (5)$ указаны номера элементов массива Z, в которых накапливаются соответствующие "коэффициенты" произведения.

2.Жирным шрифтом показаны окончательные суммы "коэффициентов," получающиеся в результате выполнения алгоритма и "накопленные" в соответствующих элементах массива Z.

3. Канторовский порядок пар элементарных многочленов выбран для удобства табличного представления. Он может быть реализован при незначительном изменении описания алгоритма.

i	j	$U_i \times V_j = Z[i+j] + Z[i+j+1]x^8 =$	$U[i]$	$V[j]$	$Z[i+j]$	$Z[i+j+1]$
0	0	$(1+x^3)\cdot x^6 = x^6 + x\cdot x^8$	10010000	00000010	(0) 00000010 **00000010**	(1) 01000000
0	1	$(1+x^3)\cdot 0 = 0$	10010000	00000000	(2) 00000000	(1) 00000000
1	0	$(1+x^2)\cdot x^6 = x^6 + x\cdot x^8$	10100000	00000010	01000000	00000010 **01000010**
0	2	$(1+x^3)\cdot(1+x+x^7) = (1+x+x^3+x^4+x^7)+x^2\cdot x^8$	10010000	11000001	(2) 11011001	(3) 00100000
1	1	$(1+x^2)\cdot 0 = 0$	10100000	00000000	00000000 **10011001**	00000000
1	2	$(1+x^2)\cdot(1+x+x^7) = (1+x+x^2+x^3+x^7)+x\cdot x^8$	10100000	11000001	(4) 01000000 **0100000**	(3) 11110001 **11010001**
1	2					

Преимущества такого алгоритма состоят в следующем. Оценим объем памяти, используемой для хранения матрицы T при $k = 8$. Таблица T содержит 2^{16} элементов, каждый из которых занимает 2 байта. Поэтому для хранения матрицы T нужно 2^{17} байт или 128 Kb, что составляет вполне разумный объем памяти для современных вычислительных средств.

С другой стороны, реализация умножения двух многочленов степени не выше $2^8 - 1 = 255$ обычным "школьным" алгоритмом приведет к не более чем $(\frac{2^8}{k})^2 = 2^{10}$ обращениям к таблице T и не более чем 2^{10} байтовым логическим сложениям, в то время, как число операций без использования матрицы T оценивается как $(2^8)^2 = 2^{16}$.

2.1.4 Модификация классического алгоритма и гибридный алгоритм умножения

Рассматриваемый в настоящем параграфе алгоритм играет вспомогательную роль в построенной нами схеме вычислений. Он используется лишь когда один из перемножаемых многочленов содержит небольшое число (не более трети) единичных коэффициентов.

Сдвигом вектора $(c_{n'-1}, c_{n'-2}, \ldots, c_1, c_0)$ будем называть вектор $(c_{n'-1}, c_{n'-2}, \ldots, c_1, c_0, 0)$.

Рассмотрим два многочлена $f(x)$ и $g(x)$ в стандартном (полиномиальном) базисе:

$$f(x) = \sum_{i=0}^{n-1} a_i x^i, \qquad g(x) = \sum_{i=0}^{n-1} b_i x^i.$$

Пусть векторы

$$A = (a_{n-1}, a_{n-2}, \ldots, a_1, a_0) \text{ и } B = (b_{n-1}, b_{n-2}, \ldots, b_1, b_0)$$

их коэффициентов заданы последовательностями машинных слов

$$C^{(k-1)}, C^{(k-2)}, \ldots, C^{(1)}, C^{(0)} \text{и} D^{(k-1)}, D^{(k-2)}, \ldots, D^{(1)}, D^{(0)}$$

длины s соответственно.

Определим последовательность векторов $V_i, \quad i = 0, 1, \ldots, s-1$ так, что $V_0 = A$, и для любого $i, \quad i = 1, 2, \ldots, n-1$, вектор V_i получается сдвигом вектора V_{i-1}. Из вектора V_i добавлением слева $(k+1)s - n - i$ нулей получим вектор V_i', с $(k+1)s$ компонентами, $i = 0, 1, \ldots, s-1$. V_i' может быть задан последовательностью $E_i = E_i^{(k)}, E_i^{(k-1)}, \ldots, E_i^{(1)}, E_i^{(0)}$ из $k+1$ машинного слова.

Пусть нужно перемножить многочлены $f(x)$ и $g(x)$. Сначала построим последовательности E_i машинных слов $i = 0, 1, \ldots, s-1$. Далее определим индексы $t_0, t_1, \ldots, t_m$ компонент вектора B, равных единице, $0 \le t_0 < t_1 < \ldots < t_m \le n-1$. Для каждого $j, \quad j = 0, 1, \ldots, m$ определим частное q_j и остаток r_j от деления числа t_j на s.

Последовательность из $2n$ нулевых машинных слов обозначим P. Таким образом, $P = L_0, L_1, \ldots, L_{2n-1}$ и $L_i = 0 \quad i = 0, 1, \ldots, 2n-1$.

Далее для каждого j, $\quad j = 0, 1, \ldots, m$ положим

$$L_{q_j+i} := L_{q_j+i} + E^{(i)}_{r_j}, \quad i = 0, 1, \ldots, k.$$

Полученная в результате описанных выше действий последовательность P машинных слов задает вектор коэффициентов произведения многочленов $f(x)$ и $g(x)$.

Теперь поясним, как ускорить поиск индексов единичных элементов вектора B. Предварительно (до начала выполнения алгоритма умножения многочленов) составим одномерную таблицу T, с помощью которой по заданному байту b определяются места единичных бит в этом байте. Перемножая многочлены $f(x)$ и $g(x)$, для каждого j-го байта машинного слова $D^{(i)}$ из последовательности $D^{(k-1)}, D^{(k-2)}, \ldots, D^{(1)}, D^{(0)}$ по таблице T определим список номеров мест единичных бит. По этим номерам и числам i, j определяется очередная порция индексов единичных элементов вектора B. Так, если ω – один из номеров в списке единичных бит j-го байта машинного слова $D^{(i)}$, то ему соответствует индекс $si + 8j + \omega$.

Пример. Пусть требуется перемножить многочлены $f(x) = x^{10} + x^8 + x^7 + x^5 + x^3 + x^2 + 1$ и $g(x) = x^8 + x^7 + x^5 + x$. В демонстрационных целях предположим, что $s = 4$. Тогда многочлены $f(x)$ и $g(x)$ задаются векторами

$$A = (1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1), \; B = (0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0),$$

соответственно, или последовательностями машинных слов

$$C = 5, 10, 13, \; D = 1, 10, 2.$$

Сначала определим последовательность векторов V_0, V_1, V_2, V_3, получаемых сдвигами из вектора A, и далее дополняем слева эти векторы нулями так, чтобы получаемые вектора состояли из 16 компонент. (16 – наименьшее число, делящееся на 4 и не меньшее количества компонент в векторе V_3) :

$$V'_0 = (0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1),$$

$$V'_1 = (0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0),$$

$$V'_2 = (0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0),$$

$$V_3' = (0,0,1,0,1,1,0,1,0,1,1,0,1,0,0,0),$$

Полученные векторы задают соответственно следующие последовательности машинных слов:

$$E_0 = 0,5,10,13, \quad E_1 = 0,11,5,10,$$
$$E_2 = 1,6,11,4, \quad E_3 = 2,13,6,8.$$

Эти последовательности задают таблицу сдвигов.

В векторе B единицы расположены на местах с номерами 1, 5, 7, 8. Деля эти числа на 4 с остатком, получаем:

$$q_0 = 0, \quad r_0 = 1, \quad q_1 = 1, \quad r_1 = 1,$$

$$q_2 = 1, \quad r_2 = 3, \quad q_3 = 2, \quad r_3 = 0.$$

В заключение вычисляем результат умножения с использованием таблицы сдвигов:

	0	0	0	11	5	10
$\bigoplus$	0	0	11	5	10	0
	0	2	13	6	8	0
	0	5	10	13	0	0
	0	7	12	5	7	10

Последовательность машинных слов, записанная под чертой, соответствует вектору коэффициентов

$$(111.1100.0101.0111.1010)$$

и задает многочлен

$$x^{18} + x^{17} + x^{16} + x^{15} + x^{14} + x^{10} + x^8 + x^6 + x^5 + x^4 + x^3 + x,$$

который и является произведением рассматриваемых в настоящем примере многочленов $f(x)$ и $g(x)$.

При перемножении некоторых заданных многочленов $f(x)$ и $g(x)$ сначала выбирается один из рассмотренных алгоритмов. Если среди перемножаемых многочленов есть содержащий небольшое число единичных коэффициентов (менее некоторого порога P), то быстрее будет работать модифицированный классический алгоритм. В противном случае следует использовать другие методы, в частности, оригинальную схему, рассматриваемую

в следующем разделе. Конкретное значение порога P определяется экспериментально при многократной апробации обоих рассматриваемых алгоритмов.

Далее приводится способ быстрого подсчета числа единичных коэффициентов многочлена $f(x)$.

Заранее строится одномерный массив M с $2^{16} = 64768$ элементами: $M[0], M[1], \ldots, M[64767]$ такой, что $M[i]$ равен числу единиц в двоичном разложении числа i. Например,

$$M[1234] = M[01001011001_2] = 5.$$

Пусть многочлен $f(x)$ задан последовательностью $A_0, A_1, \ldots, A_k$ двухбайтовых чисел. Тогда сумма $M[A_0] + M[A_1] + \ldots + M[A_k]$ равна числу единичных коэффициентов многочлена $f(x)$.

2.2 Оптимизация умножения многочленов

2.2.1 Введение

Данный раздел посвящен изучению методов выполнения умножения многочленов над полем $GF(2)$, основанных на декомпозиции по методу Карацубы [8]. Показана эффективность декомпозиционного подхода к реализации этой операций при степенях полиномов в диапазоне от 100 до 1000, характерном для криптосистем с открытым ключом.

Эффективность выполнения операций умножения достигается за счет применения оригинальной декомпозиционной схемы умножения по методу Карацубы и учета специфики операндов при модификации классического алгоритма.

Используются обозначения из предыдущей лекции.

2.2.2 Умножение многочленов по методу Карацубы

В 1962 году студент МГУ Карацуба А.А. (ныне известный специалист по теории чисел, профессор МГУ) предложил метод умножения многозначных чисел, более эффективный, чем известный классический метод умножения ¡столбиком¿.

Умножение двух $2n$-значных чисел легко сводится к четырем умножениям двух n-значных чисел. По методу Карацубы достаточно трех таких умножений.

Запишем $2n$-значное число в виде

$$A_n + 10^n \cdot B_n.$$

Легко проверяется тождество

$$(A_n + 10^n \cdot B_n) \cdot (C_n + 10^n \cdot D_n) =$$

$$A_n \cdot C_n(10^n + 1) + (D_n - C_n) \cdot (A_n - B_n) \cdot 10^n + B_n \cdot D_n \cdot (10^{2n} + 10^n).$$

Подобным образом можно поступить и при умножении двух многочленов степени $2n$ в кольце полиномов над полем Галуа $GF(2)$.

Пусть многочлены $p(x)$ и q(x) степени 2n-1 представлены в виде

$$p(x) = p_1(x) + x^n \times p_2(x),$$
$$q(x) = q_1(x) + x^n \times q_2(x),$$

где p_1, p_2, q_1, q_2 – многочлены степени $n - 1$.

Тогда произведение

$$p(x) \times q(x)$$

можно представить в форме

$$p(x) \times q(x) =$$
$$= p_1(x) \times q_1(x) +$$
$$+(p_1(x) + p_2(x)) \times (q_1(x) + q_2(x))x^n +$$
$$+p_2(x) \times q_2(x)x^{2n} +$$
$$+(p_1(x) \times q_1(x))x^n +$$
$$(p_2(x) \times q_2(x))^n.$$

Как видно, использовано только три умножения вместо четырех умножений многочленов степени n. Систематическое применение этого приема соответствует рекурсивной схеме, ¡разверткой¿ которой при различных базисах рекурсии можно получить явные схемы умножения, которые рассматриваются в следующем параграфе.

2.2.3 Оптимизация операций умножения многочленов и деления многочленов с остатком. Метод Карацубы.

Будем рассматривать многочлены в виде $\sum_{i=0}^{s} Q_i(x)x^{ik}$, где $Q_i(x)$ - элементарные многочлены степени не выше $k-1$. Рассмотрим матрицу T с 2^k строками и 2^k столбцами, в которой элемент t_{ij}, находящийся на пересечении i-й строки и j-го столбца определяется следующим образом. Пусть $a_0a_1\dots a_{k-1}$ - двоичное разложение числа $i-1$, т.е. $i-1=\sum_{r=0}^{k-1} a_r 2^r$, а $b_0b_1\dots b_{k-1}$ - двоичное разложение числа $j-1$. Тогда произведение элементарных многочленов $Q^{(1)}$, $\quad Q^{(1)}=\sum_{r=0}^{k-1} a_r x^r$ и $Q^{(2)}=\sum_{r=0}^{k-1} b_r x^r$ равно $P(x)=Q^{(3)}x^k+Q^{(4)}$ для некоторых элементарных многочленов $Q^{(3)}$ и $Q^{(4)}$. Полагаем $t_{ij}=P(x)$.

Если вычислить матрицу T заранее, то умножение элементарных многочленов сводится к отысканию нужного элемента матрицы T, что ускоряет процедуру умножения многочленов.

Пример. Пусть $k=8$ и нужно перемножить многочлены $P_1(x)=(1+x^2)x^8+(x^3+1)$ и $P_2(x)=(1+x+x^7)x^{16}+x^6$. Тогда $P_1(x)P_2(x)=(1+x^2)(1+x+x^7)x^{24}+(x^3+1)(1+x+x^7)x^{16}+(1+x^2)x^6x^8+(x^3+1)x^6$. Обращаясь к заранее вычисленной таблице произведений элементарных многочленов, получаем

$$(1+x^2)(1+x+x^7)=xx^8+(1+x+x^2+x^3+x^7),$$

$$(x^3+1)(1+x+x^7)=x^2x^8+(1+x+x^3+x^4+x^7),$$

$$(1+x^2)x^6=x^8+x^6,$$

$$(x^3+1)x^6=xx^8+x^6,$$

Откуда

$P_1(x)P_2(x)=xx^{32}+(1+x+x^2+x^3+x^7)x^{24}+x^2x^{24}+(1+x+x^3+x^4+x^7)x^{16}+x^{16}+x^6x^8+xx^8+x^6=xx^{32}+(1+x+x^3+x^7)x^{24}+(x+x^3+x^4+x^7)x^{16}+(x+x^6)x^8+x^6$.

Оценим объем памяти, используемой для хранения матрицы T при $k=8$. Таблица T содержит 2^{16} элементов, каждый из которых занимает 2 байта. Поэтому для хранения матрицы T нужно 2^{17} байт или 128 Kb, что составляет вполне разумный объем памяти для современных вычислительных средств. С другой стороны, реализация умножения двух многочленов степени не выше

$2^8-1=255$ обычным "школьным" алгоритмом приведет к не более чем $(\frac{2^8}{k})^2=2^{10}$ обращениям к таблице T и не более чем 2^{10} байтовым логическим сложениям, в то время, как число операций без использования матрицы T оценивается как $(2^8)^2=2^{16}$.

С использованием предварительных вычислений можно уменьшить время работы алгоритма деления с остатком многочлена $P_1(x)$ на $P_2(x)$ следующим образом. Через d_i обозначим $\deg P_i(x), \quad i=1,2$. Пусть $d_1 \geq d_2$. Найдем элементарный многочлен $Q(x)$ такой, что $\deg(P_1(x)+Q(x)P_2(x))x^{n_1-n_2-k+1} \leq d_1-k$ при $d_1-d_2 \geq k$ или $\deg(P_1(x)+Q(x)P_2(x)) < d_2$, $d_1-d_2<k$. Для этого в случае $d_1-d_2 \geq k$ (при $d_i \geq k$) представим многочлен $P_i(x)$ в виде $Q^{(i)}(x)x^{d_i-k+1}+P^{(i)}(x), \quad \deg P^{(x)} \leq d_i-k$ или (при $d_i<k$) положим $Q^{(i)}(x)=P_i(x) \qquad i=1,2$. В случае $d_1-d_2<k$ представим многочлен $P_1(x)$ в виде $Q_1(x)x^{d_2}+P^{(1)}(x)$ с $\deg P^{(1)}(x)<d_2$ и $P_2(x)$ в виде $Q_2(x)x^{2d_2-d_1}+P^{(2)}(x) \quad \deg P^{(2)}(x) < 2d_2-d_1$ (при $2d_2-d_1 \geq 0$) или положим $Q_2(x)=P_2(x)$ (при $2d_2-d_1<0$). Ясно, что многочлен $Q(x)$ однозначно определяется по паре элементарных многочленов $Q^{(i)}(x), \quad i=1,2$. Таким образом, можно заранее вычислить таблицу такого соответствия и использовать ее при реализации алгоритма деления многочленов с остатком.

Многочлен $P(x), \quad P(x)=\sum_{i=0}^{r} a_i(x)x^{ik}$, где a_i -элементарные многочлены в дальнейшем будем обозначать через $\sum_{i=0}^{r} a_i y^i$, используя вместо x^k переменную y. Пусть заданы два многочлена $P_i(y), \quad i=1,2$ степени (по переменной y) не большей $2t+1$. Тогда найдутся многочлены $P_i^{(j)}(y), \quad j=1,2, \quad i=1,2$, степень каждого из которых не превосходит t и $P_i(y)=P_i^{(1)}(y)+y^{t+1}P_i^{(2)}(y)$. Пусть нужно вычислить произведение $P_1(y)P_2(y)$. Алгоритм Карацубы состоит в следующем. Вычислим три произведения: $A(y)=P_1^{(1)}(y)P_2^{(1)}(y), \quad B(y)=P_1^{(2)}(y)P_2^{(2)}(y), \quad C(y)=(P_1^{(1)}+P_1^{(2)})(P_2^{(1)}+P_2^{(2)})$, а затем воспользуемся тождеством $P_1(y)P_2(y)=A(y)+(C(y)+A(y)+B(y))y^{t+1}+B(y)y^{2t+2}$. Таким образом, мы свели умножение двух многочленов к трем умножениям многочленов, имеющих "почти" в два раза меньшую степень. Применим эту процедуру далее к полученным трем произведениям, пока не сведем задачу к умножению элементарных многочленов. Сложность по времени алгоритма Карацубы составляет $O(n^{\log_2 3})$, что по порядку меньше, чем сложность "школьного" алгоритма умножения $(O(n^2))$.

Пример. Проиллюстрируем алгоритм Карацубы на примере многочленов переменной x. Пусть нужно перемножить многочлены $P_1(x) = 1 + x + x^3$ и $P_2(x) = 1 + x^2 + x^3$. Положим $P_1^{(1)}(x) = 1 + x$, $P_1^{(2)}(x) = x$, $P_2^{(1)}(x) = 1$, $P_2^{(2)}(x) = 1 + x$. Тогда $P_i(x) = P_i^{(1)}(x) + x^2 P_i^{(2)}(x)$, $i = 1, 2$, Поэтому $P_1(x)P_2(x) = (1+x)(1) + ((1+x+x)(1+1+x) + (1+x)(1) + x(1+x))x^2 + x(1+x)x^4$. Из трех полученных произведений $(1+x)1$, $1x$, $x(1+x)$ в дальнейшем рассмотрении нуждается только $x(1+x)$, которое по алгоритму Карацубы представляется в виде $0 \cdot 1 + ((0+1)(1+1) + 0 \cdot 1 + 1 \cdot 1)x + 1 \cdot 1x^2$.

2.2.4 Умножение ¡длинных¿ целых чисел

Рассмотрим способ умножения ¡длинных¿ целых чисел, который предусматривает разбиение сомножителей на байты и использование таблиц умножения байт.

Пусть ¡длинное¿ целое число L состоит из 32-х бит, т.е. из четырех байт. На этапе, предшествующем выполнению умножений ¡длинных¿ целых чисел, строится таблица T, состоящая из $2^8 = 256$ строк и такого же количества столбцов. Элемент t_{ij} таблицы T, находящийся в i-ой строке и j-ом столбце этой таблицы, $i = 0, 1, \ldots, 255$, $j = 0, 1, \ldots, 255$, представляет собой два байта, определяемых следующим образом. Пусть $i = \sum_{k=0}^{7} a_k 2^k$ и $j = \sum_{k=0}^{7} b_k 2^k$, $a_k, b_k \in \{0, 1\}$, двоичные разложения чисел i и j соответственно.

Пусть $\sum_{k=0}^{15} c_k x^k$ – результат умножения многочлена $\sum_{k=0}^{7} a_k x^k$ на многочлен $\sum_{k=0}^{7} b_k x^k$. Тогда пара байт $C_0 = c_0 c_1 \ldots c_7$ и $C_1 = c_8 c_9 \ldots c_{15}$ составляет элемент t_{ij}.

Пусть заданы два ¡длинных¿ целых числа A и B, каждое из которых состоит из четырех байт:

$$A = A[0]A[1]A[2]A[3],\ B = B[0]B[1]B[2]B[3].$$

Используя таблицу T, построенную нами для умножения байт,

сначала найдем следующие числа:

$$O_i = A[i] \times B[i], \quad i = 0,1,2,3,$$
$$T_{i,1} = (A[2i] + A[2i+1]) \times (B[2i] + B[2i+1]), \ i = 0,1,$$
$$T_{i,2} = (A[i] + A[i+2]) \times (B[i] + B[i+2]), \ i = 0,1,$$
$$F = (A[0] + A[1] + A[2] + A[3]) \times$$
$$\times (B[0] + B[1] + B[2] + B[3]).$$

Каждое из чисел $O_i, \quad i = 0,1,2,3, \quad T_{i,1}, T_{i,2}, \quad i = 0,1, \quad F$ задается парой байт. Таким образом, приведенные формулы определяют значения байт

$$O_i[0], O_i[1] \quad i = 0,1,2,3,$$

$$T_{i,1}[0], T_{i,1}[1], \ T_{i,2}[0], T_{i,2}[1] \quad i = 0,1,$$

$$F[0], F[1].$$

Используя эти числа, находим следующие величины:

$$D[1] = O_1[0] + O_0[1], \ S[1] = O_0[0] + D[1],$$
$$D[i] = O_{i(\mathrm{mod}4)}[0] + O_{i-1(\mathrm{mod}4)}[1], \ S[i] = S[i-1] + D[i],$$
$$i = 2,3,\ldots,6.$$

Если последовательность байт

$$P[0]P[1]P[2]P[3]P[4]P[5]P[6]P[7]$$

задает искомое произведение P многочленов, то

$$P[0] = O_0[0],$$
$$P[1] = S[1] + T_{0,1}[0],$$
$$P[2] = S[2] + T_{0,1}[1] + T_{0,2}[0],$$
$$P[3] = S[3] +$$
$$+T_{0,1}[0] + T_{1,1}[0] + T_{0,2}[0] + T_{0,2}[1] + T_{1,2}[0] + F[0],$$
$$P[4] = S[4] +$$
$$+T_{0,1}[1] + T_{1,1}[1] + T_{0,2}[1] + T_{1,2}[0] + T_{1,2}[1] + F[1],$$
$$P[5] = S[5] + T_{1,1}[0] + T_{1,2}[1],$$
$$P[6] = S[6] + T_{1,1}[1],$$
$$P[7] = O_3[1].$$

2.2.5 Декомпозиционная схема умножения многочленов

Рассмотрим способ построения схем умножения, с учетом размерностей сомножителей при этом будем использовать функцию умножения ¡длинных¿ (s-разрядных) целых чисел, реализация которой описана в предыдущем параграфе. Таким образом, предполагаем, что для любых двух ¡длинных¿ целых чисел A и B определен результат их умножения $S = A \times B$, коэффициенты которого задаются последовательностью из двух ¡длинных¿ целых чисел $S^{(0)}, S^{(1)}$.

На вход алгоритма поступают две последовательности, каждая из которых состоит из k ¡длинных¿ целых чисел, количество разрядов s которых совпадает с разрядностью используемого для вычислений процессора.

Схема вычислений по рассматриваемому алгоритму содержит два вспомогательных уровня и уровень вычисления результата.

На первом вспомогательном уровне вычисляются некоторые суммы ¡длинных¿ целых чисел, поступающих на вход алгоритма. Получаются новые ¡длинные¿ целые числа, которые перемножаются на этом уровне определенным образом. Каждое из полученных чисел многократно используется при дальнейших вычислениях.

Результаты умножений поступают на вход следующего уровня схемы, где путем рациональных суммирований находятся новые ¡длинные¿ целые числа. На последнем уровне выполняются также только сложения таких чисел.

Таким образом, декомпозиция операции умножения имеет вид $\Sigma\Sigma(\Pi\Sigma)$, то есть сначала выполняются предварительные сложения, потом – умножения длинных цепочек, далее – сложения с целью оптимизации вычислений с многократно используемыми числами и, наконец, – вычисления результата, использующие только сложения.

Пример. Случай $n = 6$ и $s = 32$.

Рассматриваемый при выбранных значениях n и s вариант схемы умножения позволяет перемножить многочлены степени не большей $n \cdot s - 1 = 191$.

Пусть на вход алгоритма подаются две последовательности из

шести целых чисел:

$$A^{(0)}, A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)}, A^{(5)},\ B^{(0)}, B^{(1)}, B^{(2)}, B^{(3)}, B^{(4)}, B^{(5)},$$

задающих соответственно многочлены $f(x)$ и $g(x)$. Сначала определим следующие произведения целых чисел (этапы $\Pi\Sigma$ в структурной схеме $\Sigma\Sigma(\Pi\Sigma)$)

$$\begin{gathered}
O_i = A^{(i)} * B^{(i)}, \quad i = 0, 1, \ldots, 5, \\
T_{1,i} = \left(A^{(2i)} + A^{(2i+1)}\right) * \left(B^{(2i)} + B^{(2i+1)}\right), \quad i = 0, 1, 2 \\
T_{2,i} = \left(A^{(i)} + A^{(i+2)}\right) * \left(B^{(i)} + B^{(i+2)}\right), \quad i = 0, 1 \\
T_{3,i} = \left(A^{(i)} + A^{(i+4)}\right) * \left(B^{(i)} + B^{(i+4)}\right), \quad i = 0, 1, \\
F_{1,0} = \left(A^{(0)} + A^{(1)} + A^{(2)} + A^{(3)}\right) * \\
\left(B^{(0)} + B^{(1)} + B^{(2)} + B^{(3)}\right), \\
F_{2,0} = \left(A^{(0)} + A^{(1)} + A^{(4)} + A^{(5)}\right) * \\
\left(B^{(0)} + B^{(1)} + B^{(4)} + B^{(5)}\right), \\
F_{3,i} = \left(A^{(i)} + A^{(i+2)} + A^{(i+4)}\right) * * \\
\left(B^{(i)} + B^{(i+2)} + B^{(i+4)}\right), \quad i = 0, 1, \\
E = \left(A^{(0)} + A^{(1)} + A^{(2)} + A^{(3)} + A^{(4)} + A^{(5)}\right) * \\
\left(B^{(0)} + B^{(1)} + B^{(2)} + B^{(3)} + B^{(4)} + B^{(5)}\right).
\end{gathered}$$

Так как каждое из чисел

$$O_i,\ T_{1,i},\ T_{2,i},\ T_{3,i},\ F_{1,0},\ F_{2,0},\ F_{3,i},\ E$$

задается последовательностью из двух целых s-разрядных чисел, то определены целые числа

$$O_i^{(0)}, \quad O_i^{(1)}, \quad i = 0, 1, 2, \ldots, 5,$$

$$T_{1,i}^{(0)}, \quad T_{1,i}^{(1)}, \quad i = 0, 1, 2,$$

$$T_{j,i}^{(0)}, \quad T_{j,i}^{(1)}, \quad j = 2, 3, \quad i = 0, 1,$$

$$F_{j,0}^{(0)}, \quad F_{j,0}^{(1)}, \quad j = 1, 2,$$

$$F_{3,i}^{(0)}, \quad F_{3,i}^{(1)}, \quad i = 0, 1,$$

$$E^{(0)}, \quad E^{(1)}.$$

Используя эти числа, находим следующие величины:

$$
\begin{gathered}
S^{(1)} = O_0^{(0)} + O_1^{(0)} + O_0^{(1)}, \\
S^{(i)} = S^{(i-1)} + O_i^{(0)} + O_{i-1}^{(1)}, \quad i = 2, 3, \ldots, 5, \\
S^{(6)} = S^{(5)} + O_2^{(0)} + O_4^{(0)} + O_5^{(1)}, \\
S^{(7)} = S^{(6)} + O_3^{(0)} + O_5^{(0)} + O_2^{(1)} + O_4^{(1)}, \\
S^{(8)} = O_0^{(1)} + O_1^{(1)} + O_1^{(0)} + O_4^{(0)}, \\
S^{(9)} = O_1^{(1)} + O_4^{(1)} + O_4^{(0)} + O_5^{(0)}, \\
S^{(10)} = O_4^{(1)} + O_5^{(1)} + O_5^{(0)},
\end{gathered}
$$

$$
\begin{gathered}
R^{(3)} = T_{1,0}^{(0)} + T_{1,1}^{(0)}, \\
R^{(4)} = T_{1,0}^{(1)} + T_{1,1}^{(1)}, \\
R^{(5)} = R^{(3)} + T_{1,2}^{(0)}, \\
R^{(6)} = R^{(4)} + T_{1,2}^{(1)},
\end{gathered}
$$

$$
\begin{gathered}
K^{(3)} = K^{(7)} = T_{2,0}^{(1)} + T_{2,0}^{(0)} + T_{2,1}^{(0)}, \\
K^{(4)} = K^{(8)} = T_{2,0}^{(1)} + T_{2,1}^{(0)} + T_{2,1}^{(1)},
\end{gathered}
$$

$$
\begin{gathered}
C^{(7)} = F_{1,0}^{(0)} + F_{2,0}^{(0)} + F_{3,0}^{(0)} + F_{3,1}^{(0)} + F_{3,0}^{(1)}, \\
C^{(8)} = F_{1,0}^{(1)} + F_{2,0}^{(1)} + F_{3,0}^{(1)} + F_{3,1}^{(1)} + F_{3,1}^{(0)},
\end{gathered}
$$

$$
\begin{gathered}
W^{(5)} = T_{3,0}^{(0)} + T_{3,0}^{(1)} + T_{3,1}^{(0)}, \\
W^{(6)} = W^{(5)} + T_{3,1}^{(1)}, \\
W^{(7)} = W^{(6)}, \\
W^{(8)} = W^{(7)} + T_{3,0}^{(0)},
\end{gathered}
$$

Наконец, определяем последовательность $P^{(0)}, P^{(1)}, P^{(2)}, \ldots, P^{(11)}$

целых чисел, задающую произведение многочленов $f(x)$ и $g(x)$.

$$
\begin{gathered}
P^{(0)} = O_0^{(0)}, \\
P^{(1)} = S^{(1)} + T_{1,0}^{(0)}, \\
P^{(2)} = S^{(2)} + T_{1,0}^{(1)} + T_{2,0}^{(0)}, \\
P^{(3)} = S^{(3)} + R^{(3)} + K^{(3)} + F_{1,0}^{(0)}, \\
P^{(4)} = S^{(4)} + R^{(4)} + K^{(4)} + F_{1,0}^{(1)} + T_{3,0}^{(0)}, \\
P^{(5)} = S^{(5)} + R^{(5)} + T_{2,1}^{(1)} + F_{2,0}^{(0)} + W^{(5)}, \\
P^{(6)} = S^{(6)} + R^{(6)} + T_{2,0}^{(0)} + F_{2,0}^{(1)} + F_{3,0}^{(0)} + W^{(6)}, \\
P^{(7)} = S^{(7)} + T_{1,0}^{(0)} + K^{(7)} + C^{(7)} + W^{(7)} + E^{(0)}, \\
P^{(8)} = S^{(8)} + T_{1,0}^{(1)} + K^{(8)} + C^{(8)} + W^{(8)} + E^{(1)}, \\
P^{(9)} = S^{(9)} + T_{1,2}^{(0)} + T_{2,1}^{(1)} + T_{3,1}^{(1)} + F_{3,1}^{(1)}, \\
P^{(10)} = S^{(10)} + T_{1,2}^{(1)}, \\
P^{(11)} = O_5^{(1)},
\end{gathered}
$$

Таким образом, нами построена схема умножения многочленов при $n = 6, \quad s = 32$.

Как видим, известная схема умножения многочленов по методу Карацубы представлена нами не в рекуррентном, а в явном виде. При этом разработанная схема вычислений, как уже было указано выше, имеет структуру $\Sigma\Sigma(\Pi\Sigma)$ и содержит программные эвристики.

Приведенный способ умножения многочленов допускает, очевидно, распараллеливание вычислений как внутри уровней, так и в целом.

Подобные явные схемы процедурно строятся и при бо̀льших параметрах операндов.

2.2.6 Результаты экспериментов

Осуществлена тестовая проверка следующих способов умножения многочленов в стандартном базисе:

С – классический метод умножения на уровне отдельных коэффициентов;

СТ – метод умножения на уровне элементарных многочленов степени 7 ($k = 8$) с использованием заранее вычисляемой таблицы умножения элементарных многочленов;

СМ – описанная в предыдущем параграфе модификация классического метода, минимизирующая число операций сдвига и сложения.

К – умножение многочленов по методу Карацубы (декомпозиционная схема не используется).

КС – умножение многочленов по декомпозиционной схеме.

Для получения временных экспериментальных оценок производились по 10 000 циклов умножения (с приведением результата по модулю пятичлена) с определением минимального, максимального и среднего по 10 таким испытаниям времени выполнения этих циклов. В каждом цикле осуществлялось приведение результата умножения к стандартному базису поля Галуа путем нахождения остатка от деления на неприводимый пятичлен соответствующей степени.

Результаты испытаний с использованием процессора Pentium MMX, 233 Мгц приведены в таблице (время указано в секундах). Приведенные данные получены с использованием экспериментальной библиотеки арифметических операций в конечных полях, описанной ниже в параграфе 3.3.1.

173			191			239			
Мин.	Ср.	Макс.	Мин.	Ср.	Макс.	Мин.	Ср.	Макс.	Метод
1.26	1.40	1.56	1.42	1.56	1.71	1.76	1.97	2.11	С
0.53	0.62	0.68	0.61	0.69	0.78	0.81	0.95	1.05	СТ
0.71	0.74	0.76	0.73	0.76	0.78	0.80	0.83	0.88	СМ
0.45	0.46	0.48	0.47	0.49	0.51	0.50	0.52	0.55	К
0.17	0.18	0.18	0.22	0.23	026	0.36	0.37	0.39	КС

Таким образом, выявленные сравнением теоретических оценок сложности преимущества умножения по предложенной в данной работе схеме перед классическим методом и базовым методом Карацубы подтверждаются экспериментально.

Отметим, что вычисления в соответствии с такими схемами можно осуществить также посредством логических схем.

2.3 Деление и приведение многочленов

В данной лекции изучаются способы реализации операции деления для приведения многочленов по модулю неприводимого многочлена.

2.3.1 ¡Школьный¿ алгоритм деления многочленов в стандартном базисе

С использованием операций сложения и умножения многочленов реализуется алгоритм нахождения частного и остатка при делении одного многочлена ($P_1(x)$) на другой ($P_2(x)$), т.е. нахождение таких многочленов $S(x)$ и $R(x)$, что

$$P_1(x) = S(x)P_2(x) + R(x)$$

и

$$\deg R(x) < \deg P_2(x).$$

Пример. Пусть

$$P_1(x) = x^5 + x^2 + x + 1 \text{ и } P_2(x) = x^3 + x^2.$$

Тогда

$$P_1(x) = x^2P_2(x) + x^4 + x^2 + x + 1 = x^2P_2(x) + xP_2(x) + x^3 + x^2 + x + 1 =$$

$$= x^2P_2(x) + xP_2(x) + P_2(x) + x + 1 = (x^2 + x + 1)P_2(x) + x + 1.$$

Поэтому в настоящем примере частное и остаток от деления многочлена $P_1(x)$ на многочлен $P_2(x)$ равны соответственно $S(x) = x^2 + x + 1$ и $R(x) = x + 1$.

Рассмотрим один из алгоритмов деления многочлена $P_1(x)$ на многочлен $P_2(x)$.

Даны векторы U длины p и V длины $s, s \leq p$, коэффициентов многочленов $P_1(x)$ степени $\deg P_1(x) \leq p - 1$ и $P_2(x)$ степени $s - 1$ в порядке возрастания степеней соответствующих термов многочленов, вектор U образуется двумя векторами U_1 длины $p - s$ (соответствует младшим разрядам и U_2 длины s; для формирования частного используется вектор Z длины $p - s + 1$ он образуется двумя векторами Z_1 длины 1 и Z_2 длины $p - s$ (Z_1 соответствует младшему, а Z_2 - старшим разрядам вектора Z);

Требуется вычислить частное и остаток от деления многочлена $P_1(x)$ на многочлен $P_2(x)$.

Алгоритм (так называемый "школьный" алгоритм деления) описывается следующим образом.

1. $Z = 0$,
Если $[U = 1$, то $\{Z_2 = 1, U_2 = U_2 + V$;
2. Выполнить $p - s$ раз
$U = U[\rightarrow], Z = Z[\rightarrow]$,
Если $[U = 1$ то $Z_2 = 1,\ U_2 = U_2 + V$.

Здесь $[U$ – старший элемент вектора U; $[\rightarrow]$ – операция сдвига на одну позицию в сторону старших разрядов (умножение на 2, или на полином x). Элементы вектора U_2 определяют коэффициенты многочлена-остатка; элементы вектора Z являются коэффициентами многочлена-частного.

Пример. Пусть $p = 8,\ s = 5$,

$$P_1(x) = 1 + x + x^2 + x^5,$$

$$P_2(x) = x^2 + x^3.$$

Очевидно (см. выше), что

$$S(x) = 1 + x + x^2;\ R(x) = 1 + x.$$

Вычисления представлены в следующей таблице

шаг	U_1	U_2	V	Z_1	Z_2
0	11	1001	0011	0	00
		0011		1	00
	11	**1010**		**1**	**00**
1	01	1101		0	10
		0011		1	10
	01	**1110**		**1**	**10**
2	00	1111		0	11
		0011		1	11
	00	1100		**1**	**11**

Жирным шрифтом указаны результаты каждого шага.

Сложность этого алгоритма такая же, как сложность классического алгоритма умножения.

2.3.2 Приведение многочленов по неприводимому ¡малочлену¿.

Пусть многочлен $p_1(x) = \sum_{i=0}^{n-1} u_i \times x^i$ необходимо разделить на многочлен $p_2(x) = \sum_{j=0}^{m-1} v_i \times x^i$. Частное представим многочленом $s(x) = \sum_{i=0}^{n'-1} z_i \times x^i$, а остаток – многочленом $r(x) = \sum_{i=0}^{n''-1} u'_i \times x^i$. Полагаем, что многочлен $p_2(x)$ имеет малое число ненулевых коэффициентов (является ¡малочленом¿), причем коэффициенты $v_{\deg\ p_2-1}, \ldots, v_{\deg\ p_2-s+1}$– нулевые.

Образуем список $L = [r_1, r_2, \ldots, r_l]$ степеней ненулевых коэффициентов многочлена p_2, кроме старшей.

Обозначим

$$p = \deg_s p_1,\ q = \deg_s p_2,$$

$$l_{r_i} = \left[\frac{sp - \deg\ p_2 + r_i}{s}\right], i = 1, \ldots l,$$

$$t_{r_i} = \mathrm{rest}(sp - \deg\ p_2 + r_i, s), i = 1, \ldots l.$$

Предполагается, что эти значения для каждого элемента списка L вычисляются заранее.

Двоичные последовательности коэффициентов многочленов $p_1(x)$, $p_2(x)$, $s(x)$ и $r(x)$ будем обозначать U, V, Z и U' соответственно, а их части, соответствующие разбиению на машинные слова (¡длинные¿ целые числа) будем обозначать как описано в параграфе 1.

Алгоритм вычисления многочлена-остатка $r(x)$ описывается следующим образом.

1.Пока $p > q$

а) Если $U^{(p)} \neq 0$, то для каждого элемента r_i списка L выполнить следующее:

принять $U^{(l_{r_i})} = U^{(l_{r_i})} + U^{(p_0)}$, $U^{(l_{r_i}+1)} = U^{(l_{r_i}+1)} + U^{(p_1)}$, где U^{p_0} получается из $U^{(p)}$ удалением t_{r_i} старших разрядов и добавлением t_{r_i} нулевых младших разрядов. U^{p_1} получается из $U^{(p)}$ удалением $s - t_{r_i}$ младших разрядов и добавлением $s - t_{r_i}$ нулевых старших разрядов.

б) Положить $U^{(p)} = 0$, $p = p-1$ и для каждого элемента r_i списка L принять $l_{r_i} = l_{r_i} - 1$.

2. Если $p = q$ и $\deg U^{(p)} \geq \deg V^{(q)}$

для каждого элемента r_i списка L выполнить:

из вектора $U^{(p)}$ образовать вектор W, путем "обнуления" $\deg V(q)$ младших разрядов, выполнить действия, подобные описанным в п.1 с использованием вектора W вместо вектора $U^{(p)}$ с тем отличием, что в случае $l_{r_i} < 0$ изменяется только вектор $U^{(l_{r_i}+1)}$;

"обнулить" старшие $s - \deg V^{(p)}$ разрядов вектора $U^{(p)}$.

Рассматриваемые при каждой итерации ¡длинные¿ целые числа $U^{(p)}$, $p > q$, являются числами $Z^{(p-q)}$, составляющими вектор коэффициентов частного, при этом число $Z^{(0)}$ последнего равно вектору W, получающемуся при выполнении п. 2 алгоритма. ¡Длинные¿ целые числа $U^{(j)}, j = 0, \ldots, q$ по окончании алгоритма образуют последовательность U' коэффициентов многочлена-остатка $r(x)$.

Данный вариант деления реализуется наилучшим способом при s, равном длине машинного слова $s = 32, 64$ или 128.

Для ¡малочлена¿ более общего случая (допускается один ненулевой коэффициент среди $v_{\deg\ p_2-1}, \ldots, v_{\deg\ p_2-s+1}$) при выполнении алгоритма вместо вектора $U^{(p)}$ необходимо использовать целую часть от деления многочлена $U^{(p)}(x)x^{\deg\ p_2-t}$ на двучлен $x^{\deg\ p_2-t}+1$, где t степень второго после старшего ненулевого слагаемого ¡малочлена¿, $U^{(p)}(x)$ – многочлен с последовательностью коэффициентов $U^{(p)}$.

Естественно, имеется вариант алгоритма, соответствующий разбиению последовательностей коэффициентов многочленов на байты.

Описанные в этом параграфе алгоритмы обеспечивают приведение по модулю модулярного ¡малочлена¿ за время, составляющее малую (около 10 процентов) долю времени умножения.

2.4 Возведение в степень и инвертирование в $GF(2^n)$

2.4.1 Возведение целого числа в степень по заданному модулю. Дискретный логарифм

Прототипом рассматриваемого ниже алгоритма модульного возведения многочленов в степень является соответствующий числовой алгоритм. Возведение числа a в степень e по модулю числа m можно осуществить с использованием следующего алгоритма, избегая многократного ($e-1$ раз) умножения:

1. Представить показатель e в двоичной системе счисления

$$e = e_0 2^r + \ \ldots \ + e_{r-1}2^0 + e_r,\ e_0 = 1, e_i \in \{0,1\},\ i = 1,\ \ldots,\ r.$$

2. Положить $a_0 = a$ и затем для $i = 1,\ \ldots,\ r$ вычислить

$$a_i \equiv a_{i-1}^2 \cdot a^{e_i} (\mathrm{mod}\ m).$$

3. Конец. Результатом является число a_r.

Пример 3.

Вычислим $c^{157}\ (\mathrm{mod}\ 2773)$ для $c = 1644$. Последовательно получаем

$$\begin{array}{llll} c^2 = 1834, & c^4 = 2680, & c^8 = 330, & c^{16} = 753, \\ c^{32} = 1317, & c^{64} = 1364, & c^{128} = 2586, & c^{144} = 612, \\ c^{152} = 1304, & c^{156} = 2022, & c^{157} = 2114. & \end{array}$$

Здесь при вычислении последних четырех значений в качестве операндов модульного умножения использованы ранее вычисленные степени c^{128} и c^{16}, c^{144} и c^8, c^152 и c^4 по модулю 2773 числа $c = 1644$. Наконец, модульная степень c^{157} получена умножением $c = 1644$ и $c^{156} = 2022$ по модулю 2773.

Сложность алгоритма оценивается величиной $\mathcal{O}(\ln m)$.

В отличие от этого полиномиальный алгоритм решения сравнения,

$$b^x \equiv a (\mathrm{mod}\ m),$$

то есть вычисления *дискретного логарифма x при основании b*, не известен.

2.4.2 Имплементация возведения многочленов в степень и их инвертирования

Имплементация возведения в степень 2^n.

Для нас особый интерес представляют алгоритмы инвертирования и алгоритмы возведения в степень.

Для ускорения выполнения операции возведения в квадрат в стандартном базисе мы заранее составляем таблицу, в которых для каждого байта имеются двухбайтовые слова, полученные вставкой дополнительных нулей между битами

При возведения в квадрат с использованием построенной таблицы мы вставляем между битами, задающими элемент поля (т.е. многочлен), нули и затем приводим полученный результат по модулю неприводимого многочлена, который порождает данное поле.

Отметим, что возведение в квадрат в нормальном базисе осуществляется просто циклическим сдвигом коэффициентов полинома.

Многократным повторением подобного преобразования получают степень $f(x)^{2^m}$ данного многочлена.

Имплементация инвертирования в стандартном или нормальном базисах

Рассмотрим модификацию метода возведения многочленов в степень и их инвертирования в стандартном или нормальном базисах, отличающуюся тем, что используется не рекурсия, а явные вычисления, основанные на двоичном разложении степени и машинном представлении многочленов. Описанная здесь реализация использует также некоторые программные эвристики.

Пусть многочлен $f(x)$ степени меньшей t задан в стандартном S или нормальном N базисе. Требуется получить представление многочлена $f^{-1}(x)$ в заданном базисе B, $\quad B = S$ или $B = N$.

Из равенств

$$f^{-1}(x) = f^{2^t-2}(x) = \left(f^{2^{t-1}-1}\right)^2(x)$$

следует, что для вычисления $f^{-1}(x)$ можно сначала найти $g(x) = f^{2^{t-1}-1}$, а затем возвести полученный многочлен $g(x)$ в квадрат.

Опишем часть алгоритма, касающуюся возведения многочлена $f(x)$ в степень 2^t-1, где t - натуральное. Воспользуемся равенством

$$2^t-1=\begin{cases}\left(2^{t/2}-1\right)\left(2^{t/2}+1\right), \\ \text{если} \quad t \quad \text{чётно}, \\ 2^{t-1}+\left(2^{(t-1)/2}-1\right)\left(2^{(t-1)/2}+1\right), \\ \text{если} \quad t \quad \text{нечетно}.\end{cases}$$

Отсюда

$$2^t-1=\begin{cases}\left(2^{t/2}-1\right)2^{t/2}+\left(2^{t/2}-1\right), \\ \text{если } t \quad \text{чётно}, \\ 2^{t-1}+\left(2^{(t-1)/2}-1\right)2^{(t-1)/2}+\left(2^{(t-1)/2}-1\right), \\ \text{если } t \quad \text{нечетно}.\end{cases}$$

Таким образом, для f^{2^t-1} получаем разложение

$$f^{2^t-1}=\begin{cases}\left(f^{2^{t/2}-1}\right)^{2^{t/2}}\cdot f^{2^{t/2}-1}, \\ \text{если} \quad t \quad \text{чётно}, \\ f^{2^{t-1}}\cdot\left(f^{2^{(t-1)/2}-1}\right)^{2^{(t-1)/2}}\cdot f^{2^{(t-1)/2}-1}, \\ \text{если} \quad t \quad \text{нечетно}.\end{cases}$$

Покажем, как с использованием приведенных выше разложений вычислить f^{2^t-1}, если

$$t=\sum_{i=0}^{s}a_i2^i.$$

Старший разряд a_s числа t равен 1. Пусть для некоторого s', $s'<s$, уже вычислен многочлен

$$f^{2^{t'}-1},$$

где

$$t'=\sum_{i=0}^{s'}a_{s-s'+i}2^i.$$

Найдем многочлен $f^{2^{t''}-1}$ для

$$t''=\sum_{i=0}^{s'+1}a_{s-s'-1+i}2^i.$$

Для этого рассмотрим два случая.

Случай 1. Пусть $a_{s-s'-1} = 0$. Тогда $t'' = 2t'$ и, учитывая приведенное выше разложение, получаем

$$f^{2^{t''}-1} = \left(f^{2^{t'}-1}\right)^{2^{t'}} \cdot f^{2^{t'}-1}.$$

Таким образом, нужно использовать один раз алгоритм возведения в степень для вычисления $\left(f^{2^{t'}-1}\right)^{2^{t'}}$ и один раз алгоритм умножения.

Случай 2. Пусть $a_{s-s'-1} = 1$. Тогда $t'' = 2t' + 1$ и, используя приведенное выше разложение, получаем

$$f^{2^{t''}-1} = \left(f^{2^{t'}-1}\right)^{2^{t'}} \cdot f^{2^{t'}-1} \cdot f^{2^{2t'}}.$$

Следовательно, нужно использовать два раза алгоритм возведения в степень для вычисления многочленов $\left(f^{2^{t'}-1}\right)^{2^{t'}}$, $f^{2^{2t'}}$, а затем два раза алгоритм умножения.

Выполняя описанные выше действия, начиная с $s' = 0$, чему соответствует $t' = a_s$, мы вычислим f^{2^t-1} с использованием $n(t) + b(t) - 2$ умножений, где $b(t)$, как указывалось выше, - число разрядов в двоичном разложении числа t, а $n(t)$ - число единичных разрядов в этом разложении. Таким образом порядок числа умножений равен $O(\log t)$. Кроме того, понадобится $n(t) + b(t) - 2$ раз использовать алгоритм возведения в степень.

Рассмотренный выше способ возведения многочлена f в степень $2^t - 1$ с использованием двоичного представления числа t, $t = (a_s, \ldots a_1, a_0)$, $a_s = 1$, $s > 0$. можно оформить в виде следующего алгоритма.

1.Принять $\varphi = f$.

2.Для $s' = 0, s - 1$ вычислить

$t' = \sum_{i=0}^{s'} a_{s-s'-i} \cdot 2^i$.

Если $a_{s-s'-1} = 0$ принять

$\varphi = \varphi^{2^{t'}} \cdot \varphi$,

иначе принять

$\varphi = \varphi^{2^{t'}} \cdot \varphi \cdot f^{2^{t'}}$.

Алгоритм работает как в стандартном, так и в нормальном базисах.

Пример. В поле $GF(2^3)$ многочленов, рассматриваемых по модулю неприводимого многочлена $p(x) = 1 + x^2 + x^3$ вычислим многочлен $f^{2^5-1}(x)$, если $f(x) = 1 + x$.

Здесь $t = 5$, и двоичное разложение этого числа есть $t = (a_2, a_1, a_0) = (1, 0, 1)$.

1. Примем $\varphi = f = (1 + x)$,

2.1 Для $s' = 0$ имеем

$t' = a_2 = 1$,

$a_{s-s'-1} = a_{2-0-1} = a_1 = 0$, поэтому

$\varphi = \varphi^{2^1} \cdot \varphi = \varphi^2 \cdot \varphi = (1 + x^2)(1 + x) = 1 + x + x^2 + x^3 = x$.

2.2 При $s' = 1$ получаем

$t' = a_{2-1-0}2^0 + a_{2-1-1}2^1 = a_1 2^0 + a_0 2^1 = 2$.

$a_{s-s'-1} = a_{2-1-1} = a_0 = 1$, поэтому

$\varphi = \varphi^{2^{t'}} \cdot \varphi \cdot f^{2^{2t'}} =$

$= x^{2^2} \cdot x \cdot f^{2^{2\cdot 2}} = x^4 \cdot x \cdot (1 + x)^{2^4} =$

$(1 + x + x^2) \cdot x \cdot 1 + x = (1 + x^2)(1 + x) = x$.

Пусть в рассмотренном выше поле нужно вычислить многочлен $f^{-1}(x)$, где, как и прежде, $f(x) = 1 + x$. Тогда

$$f^{-1}(x) = f^{2^3-2}(x) = \left(f^{2^2-1}(x)\right)^2.$$

(для вычисления $f^{2^2-1}(x)$ принимаем $\varphi = f = (1 + x)$, далее для $s' = 0$ находим $t' = 1$, $a_{1-0-1} = a_0 = 0$, что влечет $\varphi^{2^1} \cdot \varphi = \varphi^2 \dot{\varphi} = (1 + x)^2 \cdot (1 + x) = x$.

Таким образом, $f^{-1}(x) = \varphi^2 = x^2$.

2.4.3 Быстрый алгоритм возведения в степень в конечном поле малой характеристики в случае использования стандартного базиса

Для возведения многочлена с коэффициентами из поля $GF(2)$

$$f(x) = a_0 + a_1 x + \ldots + a_{k-1} x^{k-1}$$

в степень 2^m по модулю заданного неприводимого над $GF(2)$ многочлена $q(x)$ можно воспользоваться формулой

$$f(x)^{2^m} = a_0 + a_1 x^{2^m} + \ldots + a_{k-1} x^{(k-1)2^m},$$

в которой надо заменить каждую степень $x^{2^m i}, i = 0, \ldots, k-1$ на ее остаток по модулю многочлена $q(x)$ степени k. Составим из коэффициентов этих многочленов $k \times k$ матрицу Q_m. Эту матрицу можно вычислить со сложностью $O(mk^2)$ единственный раз, так как она не зависит от $f(x)$. Умножая вектор a коэффициентов многочлена $f(x)$ на транспонированную матрицу Q_m, что можно сделать методом Лупанова [10, 12] со сложностью $O(k^2/\log k)$, получаем коэффициенты многочлена $f(x)^{2^m}$. Последовательно возводя в квадрат, можно вычислить $f(x)^{2^n}$ со сложностью $O(nk)$. Но если выбрать $m = \frac{O(k)}{\sqrt{\log k}}, s = \lceil n/m \rceil, r = n - sm < m$, то можно с помощью указанного выше способа возвести f в степень 2^m, выполнить это s раз, получив $f^{2^{ms}}$, а потом возведя тем же способом, но с помощью матрицы Q_r, в степень 2^s получаем в итоге $(f^{2^{ms}})^{2^r} = f_{2^n}$ со сложностью $O(sk^2/\log k) = \frac{O(nk)}{\sqrt{\log k}}$, причем это можно сделать для любого $n > m$, если предварительно вычислить все матрицы $Q_r, r = 1, \ldots, m$

Заметим теперь, что при n порядка 200 вместо метода Лупанова лучше использовать стандартный метод умножения матрицы на вектор, разбивая матрицу на полосы по 32 строки и заменяя булеву n на n матрицу на матрицу размера $n/32$ на n, состоящую из long-ов, и умножая последнюю матрицу на вектор, используя операции с 32-битовыми числами. Тогда сложность умножения будет равна $n^2/16$. Так как сложность возведения в квадрат в стандартном полиномиальном базисе равна $3n/8$, то при вычислении обратного элемента в поле $GF(2^n)$ последнюю операцию возведения в $2^{(n-1)/2}$ степень выгоднее выполнить описанным выше методом (может быть и предпоследнюю тоже).

Приведем детальное описание рассмотренного алгоритма. В нем будет использованы операции над матрицами, элементами которых являются ¡длинные¿ целые числа, и операция умножения вектора на матрицу, описываемые следующим образом.

Пусть задана таблица T с k строчками и n столбцами, на пересечении i-ой строки и j-го столбца которой находится ¡длинное¿ целое неотрицательное число t_{ij}, не превосходящие $2^s - 1$, $(n-1)s < k \leq ns$. Таким образом, таблица T задает квадратную бинарную матрицу M с k строками и k столбцами, на пересечении i-ой строки и j-го столбца которой находится число a_{ij}. Введем

операцию "U" над таблицами. Результатами применения этой операции к таблице T являются таблица $U(T)$, имеющая n строк и k столбцов.

Содержательно мы преобразуем сначала таблицу T к матрице M и тем самым определяем элементы a_{ij}. Через q и r обозначим соответственно числа $]j/s[$ и $j-(q-1)s$. Тогда число a_{ij} равно остатку от деления $[t_{iq}/2^{r-1}]$ на 2.

Определим таблицу $U(T)$, на пересечении i-ой строки и j-го столбца которой находится целое неотрицательное число u_{ij}, для каждого j, $1 \le j \le k$ и для каждого i, $1 \le i \le n$ полагая

$$u_{ij} = \sum_{i'=(i-1)s}^{\min(is-1,k)} a_{i'j} 2^{i'-(i-1)s}.$$

Далее, пусть $v = (t_1, t_2, \ldots, t_n)$, где t_i - целое неотрицательное число не превосходящее $2^s - 1$, $i = 1, 2, \ldots, n$. Для каждого j, $j = 1, 2, \ldots, k$ положим

$$v'_j = \sum_{i=1}^{n} t_i u_{ij},$$

где произведение целых неотрицательных чисел t_i и u_{ij} осуществляется покомпонентно, т.е. если $t_i = \sum_{i_1=0}^{s-1} a_{i_1} 2^{i_1}$, $u_{ij} = \sum_{i_1=0}^{s-1} b_{i_1} 2^{i_1}$, то $t_i u_{ij} = \sum_{i_1=0}^{s-1} a_{i_1} b_{i_1} 2^{i_1}$; суммируются этих произведений производится также покомпонентно по модулю 2. В полученном векторе $v' = (v'_1, v'_2, \ldots, v'_k)$ каждая компонента v'_j является целым неотрицательным числом не превосходящим $2^s - 1$. Положим $c_j = 0$, если число единиц в двоичном разложении v'_j четно и $c_j = 1$ в противном случае. Вектор $c = (c_1, c_2, \ldots, c_k)$ преобразуем в вектор $u = (u_1, u_2, \ldots, u_n)$, положив

$$u_j = \sum_{i_1=s(j-1)+1}^{\min(sj,k)} c_{i_1} 2^{i_1-s(j-1)-1}.$$

Полученный вектор u является результатом умножения вектор-строки v на таблицу T.

Возведение многочлена $f(x)$ в степень 2^m непосредственно в стандартном базисе можно выполнить следующим образом.

1. Последовательно для $i = 0, 1, \ldots, k-1$ найти разложение многочлена x^{i2^m} в стандартном базисе. Так, если разложение $g_{i,m-1}(x)$ для $x^{i2^{m-1}}$ уже найдено, то искомое разложение в стандартном базисе для одночлена $x^{i2^m} = g^2_{i,m-1}(x)$ получаем, вычислив остаток от деления многочлена $g^2_{i,m-1}(x)$ на многочлен $p(x)$.

2. Построить $k \times k$-матрицу M из нулей и единиц, i-я строка которой совпадает с набором

$$a_{i-1,k-1}, a_{i-1,k-2}, \ldots, a_{i-1,1}, a_{i-1,0}$$

коэффициентов многочлена $g_{i-1,m}(x)$,

$$g_{i-1,m}(x) = \sum_{j=0}^{k-1} a_{i-1,j} x^j.$$

3. По матрице M построить таблицы T_m и $U_m = U(T_m)$.

4. Для того, чтобы получить набор коэффициентов многочлена $f^{2^m}(x)$ в стандартном базисе, нужно, представив набор коэффициентов многочлена $f(x)$ в виде $]\frac{k}{s}[$-вектора с элементами из множества $\{0, 1, \ldots, 2^s - 1\}$ и умножить его на таблицу U_m.

Для реализации этой последовательности действий понадобится:

1. Асимптотически $C_1 k^2 m$ операций для построения матрицы M, где C_1 - некоторая константа.

2. Асимптотически $C_2 k^2$ операций для построения таблицы U_m по известной матрице M, где C_2 - константа.

3. Асимптотически $C_3 \frac{k^2}{s}$ операций над s - разрядными числами для умножения вектор-строки на таблицу U_m.

Таким образом, всего понадобится выполнить асимптотически

$$C_1 k^2 m + C_2 k^2 + C_3 \frac{k^2}{s}$$

операций.

Если заранее вычислить таблицу U_m, то количество операций во время вычислений снизится до $C_3\frac{k^2}{s}$, правда необходимо будет помнить таблицы $U_m, \quad m = 1, 2, \ldots, k-1$, что потребует сохранения асимптотически k^3/s чисел.

Ниже мы покажем, как можно сократить объем требуемой памяти, уменьшив скорость работы алгоритма. Эта скорость все же будет больше, чем в случае, когда вычисления каждый раз начинаются с построения таблицы $U(T)$.

Пусть c – некоторое натуральное число (параметр), значение которого будет выбрано позже. Через m_0 обозначим число $]k/c[$. Заранее вычислим таблицы $U_m, \quad m = 1, 2, \ldots, m_0$. Объем требуемой памяти для запоминания массива $U_m, \quad m = 1, 2, \ldots, m_0$ асимптотически в c раз меньше, чем для запоминания массива $U_m, \quad m = 1, 2, \ldots, k-1$.

Пусть нужно вычислить x^{2^n} в стандартном базисе. Если $n \geq k$, то найдем остаток n_1 от деления n на k,

$$n = qk + n_1, \quad 0 \leq n_1 < k.$$

Ввиду равенства $x^{2^n} = x^{2^{n_1}}$, нужно вычислить $x^{2^{n_1}}$.

Таким образом, в дальнейшем достаточно указать способ вычисления многочлена x^{2^n} при $n < k$.

Если $n \leq m_0$, то искомый многочлен получим умножая вектор-строку, соответствующую $f(x)$ на U_n. При этом будет использовано асимптотически $C_3\frac{k^2}{s}$ операций. В противном случае через r обозначим остаток от деления числа n на m_0. Для некоторого натурального числа q_1 выполнено $n = q_1 m_0 + r$. Пусть $f_i(x), \quad i = 0, 1, \ldots, q_1, q_1 + 1$ – последовательность многочленов, представленных в стандартном базисе и полученных по следующей схеме.

$$f_0(x) = f(x),$$
$$f_i(x) = f_{i-1}^{2^{m_0}}(x), \quad i = 1, 2, \ldots, q_1.$$
$$f_{q_1+1}(x) = f_{q_1}^{2^r}(x).$$

Используя эту схему, многочлен $f^{2^n}(x) = f_{q_1+1}(x)$ можно получить, выполнив асимптотически не более

$$C_3(q_1+1)\frac{k^2}{s} \leq C_3\left(\frac{n}{m_0}+1\right)\frac{k^2}{s} \asymp C_3\frac{nkc}{s}$$

операций.

Следовательно, выбрав значение параметра $c = [\sqrt{s}]$, получаем, что для возведения многочлена $f(x)$ в степень 2^n потребуется асимптотически $C_3 \frac{nk}{\sqrt{s}}$ операций.

Пример. Пусть $k = 5, \quad p(x) = x^4+x+1, \quad f(x) = x^3+1, \quad m_0 = 2$ и требуется найти $f^{2^3}(x)$, выразив его в стандартном базисе. Сначала представим в стандартном базисе одночлены $\{1, x^2, x^4, x^6\}$. Первые два одночлена из этого множества преобразовывать не нужно;

$$x^4 = 1 + x,$$
$$x^6 = x^2 + x^3,$$

поэтому бинарную матрицу M_1, соответствующую U_1, можно представить в виде табл.1

Таблица 1.

x^3	x^2	x	x^0	
0	0	0	1	x^0
0	1	0	0	x^2
0	0	1	1	x^4
1	1	0	0	x^6

Далее представим в стандартном базисе одночленны $\{1, x^4, x^8, x^{12}\}$. Одночлен 1 преобразовывать не нужно, одночлен x^4 преобразован ранее.

$$x^8 = \left(x^4\right)^2 \equiv (1+x)^2 (\mathrm{mod} p(x)) = 1 + x^2,$$
$$x^{12} = \left(x^6\right)^2 \equiv \left(x^2 + x^3\right)^2 (\mathrm{mod} p(x)) =$$
$$= x^4 + x^6 \equiv x^3 + x^2 + x + 1 (\mathrm{mod} p(x)).$$

Отсюда бинарная матрица M_2, соответствующая U_2, имеет вид табл.2.

Таблица 2.

x^3	x^2	x	x^0	
0	0	0	1	x^0
0	0	1	1	x^4
0	1	0	1	x^8
1	1	1	1	x^{12}

Так как $3 = 1 \cdot m_0 + 1$, то вектор $(1, 0, 0, 1)$ коэффициентов многочлена $f(x)$ нужно сначала умножить на матрицу M_1. Получится вектор $(1, 1, 1, 0)$, который соответствует многочлену x^3+x^2+x. Поэтому для того, чтобы получить искомый результат, нужно вектор $(0, 1, 1, 1)$ (полученный вектор "переворачивается") умножить на матрицу M_1. Получим вектор $(1, 0, 1, 1)$, который соответствует многочлену $1 + x + x^3$. Таким образом,

$$f^{2^3}(x) \equiv 1 + x + x^3 (\mathrm{mod} p(x)).$$

2.4.4 Быстрое инвертирование в конечном поле малой характеристики с использованием стандартного базиса

Как известно, можно использовать для вычисления мультипликативного обратного в поле $GF(p^n)$ расширенный алгоритм Евклида. Время его работы возможно будет сильно зависеть от выбранного для обращения многочлена.

Можно использовать также теоретически более быстрый вариант Шенхаге-Моенка алгоритма Евклида, который дает в наихудшем случае оценку $O(M(n) \log n)$, где $M(n)$ — сложность умножения многочленов степени n

Известно также, что можно выполнять инвертирование в поле $GF(p^n)$ с помощью тождества Ферма $f^{-1} \bmod q = f^{p^n-2} \bmod q$. Используя для возведения в степень указанные выше соображения, можно получить для этого алгоритма оценку сложности $O(M(n) \log n + n^2)$. Остаточный член $O(n^2)$ можно несколько уменьшить, заменив его на $O(n^2/\sqrt{\log n})$ при условии предварительного вычисления некоторых матриц.

Для этого заметим, что для возведении f в p^n-2-ю степень приходится возводить в самом конце сразу в степень $p^{n/2+O(1)}$, до этого — в степень $p^{n/4+O(1)}$, и так далее. Если выполнять эти операции так как указано в предыдущем пункте, то суммарная сложность этих операций будет равна $\frac{O(n^2)}{\sqrt{\log n}}$ (сложность предварительных вычислений не учитываем), сложность оставшихся умножений оценивается также, как и раньше.

Глава 3

Эллиптические кривые и операции

3.1 Эллиптические кривые

3.1.1 Введение. Понятие эллиптической кривой

Эллиптические кривые применяются в криптографии с 1985 года, причем как для факторизации чисел и проверки простоты, так и для построения криптографических протоколов. Интерес к ним обусловлен, с одной стороны, тем, что они являются богатым источником конечных абелевых групп, обладающих полезными структурными свойствами, так и тем, что на их основе обеспечиваются те же криптографические свойства, которыми обладают числовые или полиномиальные криптосистемы, но при существенно меньшем размере ключа.

Эллиптической кривой E над полем $\mathcal{F}$ называется гладкая кривая, задаваемая уравнением вида

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6,\ a_i \in \mathcal{F}. \qquad (3.1)$$

Будем обозначать $\mathcal{EF}$ множество точек $(x, y) \in \mathcal{F}^2$, удовлетворяющих этому уравнению и содержащее кроме того ¡бесконечно удаленную¿ точку, обозначаемую O. Если $\mathcal{K}$ – расширение поля $\mathcal{F}$, то $\mathcal{EK}$ обозначает множество точек $(x, y) \in \mathcal{K}^2$, удовлетворяющих (3.1) вместе с точкой O. Чтобы кривая (3.1) была эллиптической кривой в $\mathcal{F}^2$ или в $\mathcal{K}^2$, она должна быть гладкой. Это означает, что в $\mathcal{F}^2$ или в $\mathcal{K}^2$ не должно быть точек, в которых равны 0 обе частные производные. Иными словами два уравнения

$$a_1 Y = 3X^2 + 2a_2 X + a_4,$$
$$2Y + a_1 X + a_3 = 0$$

не должны удовлетворяться ни в одной точке $(x, y) \in E(\mathcal{F}^2)$ или $(x, y) \in E(\mathcal{K}^2)$

С уравнением (3.1) эллиптической кривой можно связать *дискриминант* $\Delta = -16(4a^3 + 27b^2)$.

Эллиптическая кривая над полем R с ненулевым дискриминантом, $\Delta \neq 0$, представляет собой гладкую кривую, в каждой точке которой можно провести касательную.

Если $\mathcal{F}$ не является полем характеристики 2, то без потери общности можно полагать, что $a_1 = a_3 = 0$. В важном случае характеристики 2 имеется две разновидности эллиптических кривых

а) ¡суперсингулярные¿ эллиптические кривые, когда левая часть (3.1) имеет вид $Y^2 + a_3 Y$ и

б) ¡несуперсингулярные ¿ эллиптические кривые с левой частью $Y^2 + a_1 XY$, $a_1 \neq 0$, этого уравнения. Для полей характеристики 2 можно также положить $a_2 = 0$ в суперсингулярном случае и $a_4 = 0$ в несуперсингулярном случае.

Если характеристика поля не равна ни 2, ни 3, то после упрощения левой части (3.1), линейной заменой переменной (а именно, $X \to X - 1/3a_2$) можно также удалить терм X^2. То есть без потери общности можно полагать, что кривая задана уравнениями вида

$$Y^2 = X^3 + aX + b,\ a, b, \in \mathcal{F},\ \mathrm{char}\mathcal{F} \neq 2, 3, \qquad (3.2)$$

В этом случае условие гладкости кривой состоит в требовании, что кубический многочлен справа не имеет кратных корней. А это выполняется тогда и только тогда, когда его дискриминант не равен нулю.

Для геометрической интерпретации термина "эллиптическая кривая" и операций над ее точками полезно построить график эллиптической кривой

$$Y^2 = X^3 - X,$$

рассматриваемой над полем $\mathcal{R}$ действительных чисел.

Заметим, что при больших X кривая ведет себя как функция $Y = X^{3/2}$.

Отметим еще, что эллиптической кривой $Y^2 = f(X)$ соответствует *эллиптический интеграл*

$$\int \frac{dX}{\sqrt{f(X)}},$$

не берущийся в элементарных функциях. Эллиптические интегралы, в свою очередь, возникают при вычислении длин дуг эллипсов.

Например, кривой $Y^2 = X^3 - X$ соответствует *эллиптический интеграл*

$$\int \frac{dX}{\sqrt{X^3 - X}}.$$

3.1.2 Закон сложения

Для некоторого расширения $\mathcal{K}$ поля $\mathcal{F}$ множество $\mathcal{EK}$ образует абелеву группу с тождественным элементом (единицей) O. Чтобы объяснить правило сложения точек эллиптической кривой, сначала лучше рассмотреть абелеву группу эллиптической кривой, определенной над полем $\mathcal{R}$. действительных чисел.

Пусть E эллиптическая кривая над полем действительных чисел, заданная уравнением (3.2), и пусть P и Q – две точки на E. Определим противоположный элемент к P (то есть обратный элемент) и сумму $P + Q$ по следующему правилу:

1) если P есть точка O, мы определяем $-P$ как O. Для каждой точки Q мы полагаем, что $Q + O = Q$, то есть точка O выполняет роль единицы по сложению (¡нулевого¿ элемента группы точек).

2) Отрицание $-P$ есть точка с той же x-координатой, но с отрицанием y-координаты, то есть $-(x, y) = (x, -y)$. Очевидно из уравнения (3.2), что $(x, -y)$ находится на той же части кривой, что и точка x, y. Если $Q = -P$, то мы определяем сумму $Q + P$ как точку бесконечности O.

3)Если P и Q имеют различные x-координаты, то можно показать, что линия $\mathcal{L}=P\bar{Q}$ пересекает кривую точно в одной точке R. (Эта точка может совпасть с точкой P или Q, тогда прямая $\mathcal{L}$ является касательной к кривой в точке P или Q и тогда мы полагаем $R = P$ или $R = Q$ соответственно). Затем мы определяем $P+Q$

как $-R$, то есть как зеркальное относительно оси x отображение третьей точки пересечения R.

4) Последняя возможность – это $P = Q$. Пусть $\mathcal{L}$ является касательной к кривой в точке P и пусть R – единственная точка пересечения прямой с эллиптической кривой, тогда мы полагаем, что $2P = -R$. (В этом случае точка P является точкой "дублированного касания", или точкой инфлекции).

Если прямая проходит через точку O бесконечности, то это отношение представляется в форме $P + \tilde{P} + O = O$, где P и $\tilde{P}$ – симметричные точки, т.е. $\tilde{P} = -P$. В противном случае оно имеет вид $P + Q + R = O$, где P, Q и R – три точки кривой на прямой $\mathcal{L}$, соответствующие правилу 3) или 4).

Таким образом, сумма любых трех точек эллиптической кривой, лежащих на одной прямой, равна O.

Теперь мы покажем, что имеется точно одна дополнительная точка пересечения эллиптической кривой прямой, проходящей через две заданные точки P и Q кривой, и одновременно для кривой (3.2) выведем формулы для координат этой точки, а тем самым и точки $P + Q$.

Пусть $(x_1, y_1), (x_2, y_2)$ и (x_3, y_3), – обозначения координат точек P, Q и $P + Q$ соответственно.

Пусть имеется случай 3) определения $P+Q$, и пусть $y = \alpha x + \beta$ – уравнение прямой, проходящей через точки P и Q, (которая не является в случае 3) вертикальной линией). Тогда $\alpha = (y_2 - y_1)/(x_2 - x_1)$, и $\beta = y_1 - \alpha x_1$. Точка $(x, \alpha x + \beta) \in \mathcal{L}$ лежит на эллиптической кривой только в том случае, когда $(\alpha x + \beta)^2 + 2 = x^3 + ax + b$. Таким образом, имеется только одна точка пересечения для каждого корня кубического уравнения $x^3 - (\alpha x + \beta)^2 + ax + b$. Два корня x_1 и x_2 мы уже знаем, так как $(x_1, \alpha x_1 + \beta)$ и $x_2, \alpha x_2 + \beta$ являются точками P и Q кривой. Так как сумма корней нормированного многочлена равна взятому со знаком минус коэффициенту при степени, предшествующей старшей степени, мы можем заключить, что третий корень в этом случае определяется как $x_3 = \alpha^2 - x_1 - x_2$.

Это позволяет получить выражение для x_3 и, следовательно, для обеих координат суммы $P + Q = (x_3, -(\alpha x_3 + \beta))$ через коор-

динаты x_1, x_2, y_1, y_2 :

$$x_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)^2 - x_1 - x_2,$$
$$y_3 = -y_1 + \left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1 - x_3).$$

Случай, когда $P = Q$, подобен рассмотренному, за исключением того, что α теперь есть производная dy/dx в точке P. Явное дифференцирование уравнения (3.2) дает формулу $\alpha = (3x_1^2 + a)/2y_1$, и мы получаем координаты точки $2P$ при удвоении точки P :

$$x_3 = \left(\frac{3x_1^2+a}{2y_1}\right)^2 - 2x_1,$$
$$y_3 = -y_1 + \left(\frac{3x_1^2+a}{2y_1}\right)(x_1 - x_3).$$

Пример 1.1. . Пусть $P = (0, 0)$ на эллиптической кривой

$$Y^2 + Y = X^3 - X^2.$$

Найти $2P = P + P$ и $3P = P + 2P$.

Решение. Прежде всего преобразуем уравнение путем замены переменных $Y \to Y - 1/2,\ X \to X + 1/3$ к виду

$$Y^2 = X^3 - \frac{1}{3}x + \left(\frac{1}{4} - \frac{2}{27}\right).$$

На этой кривой точка P становится точкой $Q = (-1/3, 1/2)$. Используя формулы удвоения, получим $2Q = (2/3, -1/2)$. Далее вычисляем $3Q = 2Q + Q = (2/3, 1/2)$. Заметим, что $3Q = -(2Q)$, и следовательно Q является точкой порядка 5, то есть $5Q = O$. Возвращаясь к исходной кривой, имеем $2P = (1, -1),\ 3P = (1, 0) = -2P$.

Можно доказать, что операция сложения точек эллиптической кривой коммутативна и ассоциативна, то есть множество точек в месте с точкой бесконечности O образует абелеву группу. Такое доказательство получают в проективной геометрии с использованием следующего утверждения:

Утверждение 1.1. *Пусть три прямые $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ пересекают кубическую кривую в девяти точках $P_1, P_2, \ldots, P_9$ с возможными совпадениями и пусть $\mathcal{L}'_1, \mathcal{L}'_2, \mathcal{L}'_3$ – три прямые, пересекающие кривую в точках $Q_1, Q_2, \ldots, Q_9$. Если $P_i = Q_i$ для $i = 1, \ldots 8$, то также $P_9 = Q_9$.*

Как в любой абелевой группе будем использовать обозначение kP для точки, полученной k-кратным сложением точки P самой с собой ($P+\ldots+P, k$ раз) и обозначение $-kP$ для $(-P)+\ldots+(-P), k$ раз.

Пока что мы ограничиваемся представление о точке бесконечности O как о точке, расположенной бесконечно далеко в положительном направлении оси y и рассматриваемой в качестве третьей точки пересечения эллиптической кривой любой вертикальной линией, всякая такая линия пересекает кривую в точках $(x_1, y_1), (x_2, y_2), O$.

Более естественно точка бесконечности определяется в проективных координатах.

3.1.3 Проективные координаты

Проективной плоскостью над полем $\mathcal{F}$ называется множество классов эквивалентности троек (X, Y, Z), в которых хотя вы один элемент ненулевой. Эквивалентными считаются тройки, если элементы одной из них получаются умножением на скаляр элементов другой: $(X', Y', Z') \sim (X, Y, Z,)$, если для некоторого элемента $\lambda \in \mathcal{F}$ $(\lambda X', \lambda Y', \lambda Z') = (X, Y, Z)$. такие классы эквивалентности называются *проективными точками.* Проективные точки с ненулевым элементом Z принадлежат классу эквивалентности, содержащему единственную точку вида $(x, y, 1)$: она просто вычисляется $x = X/Z$, $y = Y/Z$. Таким образом, проективная плоскость может быть определена как множество всех точек (x, y) обычной ("афинной") плоскости с дополнением точек, для которых $Z = 0$. Эти точки можно интерпретировать как линию в бесконечности и рассматривать ее как "горизонт" плоскости. Всякое уравнение $F(X, Y) = 0$ кривой в афинной плоскости соответствует *однородному* уравнению $\tilde{F}(X, Y, Z) = 0$, выполняемое для соответствующих проективных точек: просто заменим X на X/Z, $y = Y/Z$ и умножим на степень Z, чтобы избавиться от знаменателей. Например, если применить эту процедуру к к афинному уравнению $Y^2 = X^3 + aX + b$ эллиптической кривой, то получится " проективное уравнение" $Y^2Z = X^3 + aXZ^2 + bZ^3$. Это уравнение выполняется для проективной тройки (X, Y, Z), $Z \neq 0$, тогда и только тогда, ко-

гда соответствующая афинная точка (x, y), где $x = X/Z,\ y = Y/Z$, удовлетворяет уравнению (3.2).

Какие ещё проективные точки (X, Y, Z), кроме точек с $Z \neq 0$, удовлетворяют уравнению $\tilde{F} = 0$? Подставляя $Z = 0$ в уравнение, получаем $0 = X^3$, это означает, что $X = 0$. Но имеется только один класс эквивалентности троек (X, Y, Z), где оба элемента X и Z нулевые – класс, содержащий $(0, 1, 0)$. Это и есть точка, которую мы обозначаем O. Это точка пересечения оси y с линией бесконечности.

3.1.4 Эллиптические кривые над полями характеристики 2 и 3

Если $\mathrm{char}(\mathcal{F}) = 2$, то эллиптическую кривую нельзя представить в форме (3.2) – действительно, кривая (3.2) не может быть гладкой над полем характеристики 2. В случае характеристики 3 нельзя удалить терм a_2X^2, если он сам по себе не нулевой. Таким образом и формулы сложения из предыдущего параграфа мы не можем использовать непосредственно.

Конечно, можно вывести аналогичные формулы применительно к эллиптическим кривым, уравнения которых имеют более общую форму (3.2), пригодную для использования при любой характеристике. Однако эти формулы выглядят громоздко.

Но для характеристик 3 и 2 все же можно получить удобные формулы.

1) Если $a_1 = a_3 = 0$ в (3.1), а a_2 не обязательно равен нулю, получим для $\mathrm{char}\mathcal{F}$=3:

$$x_3 = \left(\frac{y_2-y_1}{x_2-x_1}\right)^2 - a_2 - x_1 - x_2,$$
$$y_3 = -y_1 + \left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1 - x_3).$$

при сложении разных точек и

$$x_3 = \left(\frac{3x_1^2+2a_2x-1+a_4}{2y_1}\right)^2 - a_2 - 2x_1,$$
$$y_3 = -y_1 + \left(\frac{3x_1^2+2a_2x_1+a_4}{2y_1}\right)(x_1 - x_3),$$

при удвоении.

2) Если $a_3 = a_4 = 0$ в (1.1), а a_2 ненулевой и предполагается равным 1, то при $char\mathcal{F} = 2$ в несуперсингулярном случае имеем

$$x_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + \frac{y_1+y_2}{x_1+x_2} + x_1 + x_2 + a_2,$$
$$y_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1 + x_3) + x_3 + y_1,$$

при сложении различных точек и

$$x_3 = x_1^2 + a_6/x_1^2,$$
$$y_3 = -x_1^2 + \left(\frac{x_1+y_1}{x_1}\right) x_3 + x_3,$$

при удвоении.

3)Если $a_1 = a_2 = 0$ в (3.2), а $a_3 \neq 0$ и char$\mathcal{F}$=2 (суперсингулярный случай)

$$x_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + x_1 + x_2,$$
$$y_3 = \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1 + x_3) + y_1 + a_3,$$

при сложении различных точек и

$$x_3 = \frac{x_1^4+a_4^2}{a_3^2},$$
$$y_3 = \left(\frac{x_1^2+a_4}{a_3}\right)(x_1 + x_3) + y_1 + a_3,$$

при удвоении.

Во всех случаях правила прибавления точки $P_1 = (x_1, y_1)$ к точке $P_2 = (x_2, y_2)$ для получения точки $P_3 = P_1 + P2 = (x_3, y_3)$ могут быть записаны в виде выражений для x_3 и y_3 с использованием x_1, x_2, y_1, y_4 и коэффициентов a_i.

3.2 Эллиптические кривые над полем $GF(2^n)$

В этом разделе приводятся некоторые факты о эллиптических кривых над полем $GF(2^n)$ и алгоритмы, имеющие криптографическое значение.

3.2.1 Суперсингулярные кривые

Основное различие между двумя типами эллиптических кривых – несуперсингулярными и суперсингулярными – это то, что

для суперсингулярных кривых известен порядок (количество точек) соответствующей группы $\mathcal{EK}$ (здесь $\mathcal{K} = GF(2^n)$).

При нечетном n имеется 3 класса неизоморфных суперсингулярных эллиптических кривых (отметим, что при четном n имеется 7 классов), стандартными представителями которых являются кривые

$$\mathcal{E}_1 : y^2 + y = x^3,$$

$$\mathcal{E}_2 : y^2 + y = x^3 + x$$

и

$$\mathcal{E}_3 : y^2 + y = x^3 + x + 1.$$

При нечетном n число точек для первой кривой равно $2^n + 1$ и $2^n \pm \sqrt{2^{n+1}} + 1$ для второй и третьей (знак + или − выбирается в зависимости от кривой и от сравнения n по модулю 8). Указанные значения легко вычисляются с использованием теоремы Вейля (см. параграф 3.2.5). Отметим, что группы этих кривых при нечетном n являются циклическими.

В общем случае, группа $\mathcal{EK}$ — или циклическая (изоморфная Z_m (при некотором m), или прямая сумма двух циклических групп, $\mathcal{EK} \cong Z_{m_1} \oplus Z_{m_2}$, где m_2 делит m_1 и m_2 делит $q - 1$, $q = 2^n$.

Нас будут интересовать кривые $\mathcal{E}_2$ и $\mathcal{E}_3$ (кривую $\mathcal{E}_1$ мы использовать не будем), а для них особый интерес будут представлять такие n, для которых соответствующий порядок группы $\mathcal{EK}$ при разложении на простые множители содержит большое простое число (кстати, большое простое с N цифрами в десятичном представлении будем записывать как PN).

Ниже приведены некоторые конкретные значения, которые можно использовать для имплементации. Для n=173 неприводимые трёхчлены или пятичлены можно взять из приложениий.

n	Кривая	Порядок группы
173	$\mathcal{E}_2$	$5 \cdot 13625405957 \cdot P42$
173	$\mathcal{E}_3$	$7152893721041 \cdot P40$
191	$\mathcal{E}_2$	$5 \cdot 3821 \cdot 89618875387061 \cdot P40$
191	$\mathcal{E}_3$	$25212001 \cdot 5972216269 \cdot P41$
239	$\mathcal{E}_2$	$5 \cdot 77852679293 \cdot P61$
239	$\mathcal{E}_3$	$P72$
323	$\mathcal{E}_3$	$137 \cdot 953 \cdot 525313 \cdot P87$

3.2.2 Формулы сложения

Сложение точек в случае суперсингулярных эллиптических кривых производится по приведенным в параграфе 3.1.4 формулам.

Заметим, что при сложении точек выполняется одна инверсия, которая является самой трудоемкой операцией в арифметике конечного поля.

Для координатного представления нулевого элемента O группы $\mathcal{EK}$ фиксируем какое-нибудь координатное представление, задающую точку, не лежащую на кривой, например, $(1,1)$ для E_2 или $(0,0)$ для E_3.

3.2.3 Алгоритмы вычисления kP

Алгоритмы вычисления kP являются основными в арифметике эллиптических кривых. Хотя они вполне аналогичны уже рассмотренным алгоритмам возведения в степень в стандартных базисах конечных полей, мы рассмотрим их поподробнее.

Кроме выделения случая, когда точка P известна заранее, ускорения вычислений можно достичь, выбирая k (это будет в нашей власти) с числом единиц в диапазоне 40–60.

Альтернативный подход к ускорению вычислений связан с использованием проективных координат.

Вычисление kP – первый способ

Итак, рассмотрим алгоритм вычисления координат точки kP,

где k – натуральное число, (называемое ключом), а P – точка плоскости, задаваемая парой координат (x_0, y_0). Здесь x_0 и y_0 – многочлены из поля Галуа $GF(2^n)$. Мы предполагаем, что $P \neq 0$, т.к. в противном случае $kP = 0$ и вычислять ничего не надо. Предварительно число k разложим в двоичной системе:

$$k = \sum_{i=0}^{m-1} a_i 2^i, \qquad a_i \in \{0, 1\},$$

где $m =]\log_2(k+1)[$. Как указывалось ранее, в нашей власти выбрать число k, двоичное разложение которого содержит 40-60 единиц. Таким образом, среди чисел $a_0, a_1, \ldots, a_{m-1}$ содержится 40-60 единиц.

Пусть $i_1, i_2, \ldots, i_t$ – индексы единичных компонент в наборе

$$a_0, a_1, \ldots, a_{m-1}, \quad i_1 < i_2 < \ldots < i_{t-1}.$$

Тогда

$$kP = \left(\sum_{j=1}^{t} 2^{i_j} \right) \cdot P.$$

Выше были приведены формулы для вычисления произведения $2P$. Используя n раз эту формулу, можно получить $2^n P$.

Далее найдем последовательность $A_j = 2^{i_j} P, \quad i = 1, 2, \ldots, t$ по следующей схеме

$$A_1 = 2^{i_1} P,$$
$$A_j = 2^{i_j - i_{j-1}} A_{j-1}, \quad j = 2, 3, \ldots, t.$$

Сложив все полученные $A_j, \quad j = 1, 2, \ldots, t$, получим искомое произведение kP:

$$kP = \sum_{j=1}^{t} A_j.$$

Этот алгоритм использует не более $\log_2 t$ умножений многочленов на двойку и не более 40-60 операций сложения многочленов.

Пример. Пусть в поле $GF(4)$ с модулярным многочленом $f(x) = x^4 + x + 1$ требуется вычислить $10P$, если $P = (x_0, y_0) = \left(x^3 + x + 1, x + 1\right)$. Имеем следующее разложение числа 10 в двоичной системе счисления

$$10 = 2^3 + 2^1 = 1010_2.$$

Поэтому $t = 2, \quad i_1 = 1, i_2 = 3$. Вычислим $A_1, \quad A_1 = 2P$. Обозначим через (x_1, y_1) координаты точки A_1. Тогда, используя рассмотренные выше формулы, получаем

$$x_1 = x_0^4 + 1 = \left(x^3 + x + 1\right)^4 + 1 = x^3 + x^2,$$

$$y_1 = x_0^4 + 1 + y_0^4 = x^3 + x^2 + (x+1)^4 = x^3 + x^2 + x.$$

Далее найдем координаты (x_2, y_2) точки $A_2 = 2^2 A_1$ Для этого вычисляем

$$x_1' = x_1^4 + 1 = x^3 + x + 1,$$

$$y_1' = x_1' + y_1^4 = x^3 + x + 1 + x^3 + 1 = x,$$

откуда

$$x_2 = (x_1')^4 + 1 = \left(x^3 + 1\right)^2 + 1 = x^3 + x^2,$$

$$y_2 = x_2 + (y_1')^4 = x^3 + x^2 + x + 1.$$

Таким образом,

$$10P = A_1 + A_2 = \left(x^3 + x^2, x^3 + x^2 + x\right) + \left(x^3 + x^2, x^3 + x^2 + x + 1\right)$$

Следуя правилам сложения точек, получаем $10P = 0$, т.к. точки A_1 и A_2 имеют равные абсциссы, но разные ординаты.

О вычислении kP методом аддитивных цепочек

Вычисление кратных заданной точки эллиптической кривой можно выполнять тем же способом, что и возведение в степень в конечном поле. Опишем этот метод [2], используя аддитивную символику.

Чтобы вычислить точку $k \cdot P$, разложим k в системе счисления по основанию 2^m, именно

$$k = \sum_{i=0}^{\lfloor n/m \rfloor} a_i 2^{mi},$$

вычислим заранее все кратные $a_i P$ (в худшем случае для этого надо будет вычислить все кратные $P, 2P, \ldots, (2^m - 1)P$ с помощью поочередных удвоений и прибавлений P, что требует 2^{m-1} и тех и других операций), а потом вычисляем kP по схеме Горнера

$$kP = (\ldots(a_{s-1}2^m + a_{s-2})2^m + \ldots + a_1)2^m) + a_0)P,$$

используя $s = \lfloor n/m \rfloor$ сложения с уже вычисленными точками $a_i P$ и столько же умножений на 2^m.

В общем случае оценка сложности имеет вид

$$2^{m-1}(M+K) + Mn/m + nK,$$

где M и K — сложности сложения и удвоения точек соответственно.

Выбирая

$$m = \lfloor \log_2 n - \log_2 \log_2 n - \log_2 \log_2 \log_2 n \rfloor$$

получаем асимптотически точную в общем случае оценку сложности

$$nK + Mn/m + o((M+K)n/m).$$

Для конкретных k ее можно улучшить, увеличивая m, но при этом более аккуратно вычисляя $a_i P$, которых при удачном выборе m может оказаться гораздо меньше, чем 2^m. Можно написать программу выбора m с минимизацией количества различных a_i, причем за счет сдвига можно всегда считать, что двоичные записи чисел a_i всегда начинаются с единицы.

Приведём ещё один способ вычисления kP. Разбивая все числа a_i на группы одинаковых, можно представить kP в виде

$$\sum_{j=1}^{d} a_{i_j} \sum_{i \in M_j} 2^i = \sum_{j=1}^{d} 2^{m_j} a_{i_j} b_j,$$

откуда применяя схему Горнера, получаем, что сложность kP оценивается сверху как $dM + Kn$ плюс сложность системы всех различных точек $a_i P$, плюс сумма сложностей вычисления точек $b_j P_j$. Далее проводим минимизацию по параметру m. Сложность вычисления каждой из точек $b_j P_j$ вычисляется рекурсивно, также рекурсивно вычисляется сложность вычисления системы точек $a_i P$, если разложить все эти числа по схеме Горнера с одним и тем же параметром m.

Нас интересует случай n порядка 200, тогда m можно выбирать 6,7,8 в худшем случае. Если же точка P выбирается заранее, все ее кратные $P, 2P, \ldots, (2^m - 1)P$ можно заранее занести в память, и указанная выше проблема минимизации исчезает, тогда m можно выбрать порядка 16 (чтобы упомянутые выше точки уместились в быструю кэш-память).

Применение для эллиптических кривых быстрого алгоритма возведения многочлена в степень в случае использования стандартного базиса

К вычислению кратных точек для эллиптических кривых. можно применить и алгоритм из параграфа 2.4.3

Операция удвоения точки $P = (x, y)$ эллиптической кривой $\mathcal{E}_2$ и $\mathcal{E}_3$ выполняется по формуле

$$2P = (x^4 + 1, x^4 + y^4 + 1)$$

и поэтому сводится к возведению в квадрат в конечном поле. Более того, вычисление $2^{2n}P$ можно выполнить по формуле

$$2^{2n}P = (x^{2^{4n}}, y^{2^{4n}} + 1).$$

Поэтому вычисление точки 2^nP можно выполнить быстро с использованием описанных выше процедур.

3.2.4 Использование проективных координат

Исключить операцию инверсии за счет увеличения общего числа умножений можно, переходя к проективной плоскости.

Путем подстановки

$$x = \frac{x}{z}, \qquad y = \frac{y}{z}$$

в уравнения, задающие эллиптические кривые E_2 и E_3 мы приходим к следующим однородными уравнениями относительно x, y и z:

$$E_2 : y^2z + yz^2 = x^3 + xz^2, \quad E_3 : y^2z + yz^2 = x^3 + xz^2 + z^3.$$

Наряду с обычными (аффинными) координатами мы можем теперь рассматривать проективные координаты — ненулевые тройки из K^3. Проективные координаты (x, y, z) и (x', y', z') считаются эквивалентными, если для некоторого ненулевого t из K, выполняется

$$x' = tx, \quad y' = ty, \quad z' = tz.$$

Класс эквивалентностей (а также его представитель), порожденный тройкой (x, y, z) обозначаем $(x : y : z)$. Множество всех

классов эквивалентностей и называем *проективными* координатами. Геометрически (двумерное) *проективное пространство* можно представлять себе как множество прямых в обычном трехмерном пространстве, проходящих через начало координат.

Теперь эллиптическую кривую можно рассматривать на проективной плоскости как множество проективных точек, удовлетворяющих соответствующему однородному уравнению. Заметим, что единственной проективной точкой с нулевой z координатой, лежащей на эллиптической кривой, будет точка $(0:1:0)$, которая соответствует бесконечно удаленной точке O. Для остальных точек кривой $(x:y:z) \sim (x/z:y/z:1)$, так что проективная точка $(x:y:z)$ однозначно соответствует аффинной точке $(x/z, y/z)$.

Мы хотим теперь получить формулы для сложения проективных точек P и Q.

Пусть $P = (x_1 : y_1 : 1) \in E_2(E_3)$ (таким образом, одна точка у нас фактически задана обычными координатами) и $Q = (x_2 : y_2 : z_2) \in E_2(E_3)$. Предположим, что $P, Q \neq O$ и $P \neq Q$ (нас интересует основной случай в сложении точек). Для точки $R = P + Q$, $R = (x_3' : y_3' : 1)$ мы можем использовать формулы сложения в аффинном случае, после применения которых получим:

$$x_3' = \frac{a^2}{b^2} + x_1 + \frac{x_2}{z_2},$$

$$y_3' = 1 + y_1 + \frac{a}{b}(\frac{a^2}{b^2} + \frac{x_2}{z_2}),$$

где

$$a = y_1 z_2 + y_2, \quad b = x_1 x_2 + x_2.$$

Полагая

$$z_3 = b^3 z_2, \quad x_3 = x_3' z_3, \quad y_3 = y_3' z_3$$

мы получим $R = (x_3 : y_3 : z_3)$, где

$$x_3 = a^2 b z_2 + b^4,$$

$$y_3 = (1 + y_1) z_3 + a^3 z_2 + a b^2 x_2,$$

$$z_3 = b^3 z_2.$$

Видно, что не считая сложений и возведений в степень, нам для сложения точек в проективных координатах необходимо выполнить 9 умножений (в аффинных координатах только 2), но зато ни одного обращения.

При вычислении kP мы последовательно удваиваем точки (что не требует инверсий), а затем складываем некоторые из них, накапливая результат в Q. Окончательный результат, полученный в проективных координатах, преобразуем в аффинные делением на z_3^{-1} (одна инверсия в самом конце).

3.2.5 Несуперсингулярные кривые

Вообще говоря, несуперсингулярные эллиптические кривые представляют больший интерес с криптографической точки зрения, по сравнению с суперсингулярными. Это связано с тем, что для суперсингулярных кривых известно [14] сведение задачи вычисления дискретного логарифма к аналогичной задаче для конечных полей (с повышением размерности поля), а для несуперсингулярных эллиптических кривых такого сведения пока нет.

Так для суперсингулярной эллиптической кривой $\mathcal{E}_1$, заданной над полем порядка q, задача вычисления дискретного логарифма в группе этой эллиптической кривой сводится к соответствующей залаче вычисления в поле порядка $2q$. Для кривых $\mathcal{E}_2$ и $\mathcal{E}_3$ такое сведение приводит уже к полям порядка $4q$.

Несмотря на то, что для некоторых полей в случае характеристики 2 $GF(2^n)$ известны специальные алгоритмы вычисления дискретных логарифмов (алгоритм Куперсмита и др.), тем не менее они остаются практически неосуществимыми лоя полей 500-1000 и более. Это оправдывает использование на практике кривых $\mathcal{E}_2$ и $\mathcal{E}_3$. Однако для несуперсингулярных кривых оказывается довольно сложно вычислить порядок соответствующей группы. Порядок же группы мы должны знать, чтобы уметь находить на кривой точки достаточно высокого порядка.

В простейших случаях порядок группы эллиптической кривой можно найти с использованием теоремы Вейля.

Теорема 2.1. *Пусть $\mathcal{E}$ – эллиптическая кривая над полем $\mathcal{F}_q$ и m – порядок её группы. Тогда для порядка $M(n)$ группы эллиптиче-*

ской кривой $\mathcal{EF}_{q^n}$ над любым полем $\mathcal{F}_{q^n}$, являющемся расширением поля $\mathcal{F}_q$, справедлива формула

$$M(n) = q^n + 1 - \alpha^n - \beta^n,$$

где α и β – корни квадратного уравнения $x2 - tx + q = 0$, в котором коэффициент $t = q + 1 + m$.

Заметим, что по теореме Хассе выполняется неравентсво $t^2 \leq 4q$ и в случае строгого неравенства корни квадратного уравнения α и β будут комплексными.

Пример 2.1. Найдем порядок группы эллиптической кривой в случае $\mathcal{E}_2$. Рассматривая $\mathcal{E}_2$ над полем $GF(2)$, имеем на ней точки (0,0),(0,1),(1,0),(1,1) и ещё нулевой элемент O – всего пять элементов. циклической группы. Тукам образом, $q = 2, m = 5, t = -2$, и мы находим корни квадратного уравнения

$$x^2 + 2x + 2 = 0 :$$

$\alpha = -1 + \sqrt{i}$ и $\beta = -1 - \sqrt{i}$ или в тригонометрической записи коиплексных чисел

$$\alpha = \sqrt{2}\left(\cos\frac{3\pi}{4} + i\sin\frac{3\pi}{4}\right)$$

и

$$\beta = \sqrt{2}\left(\cos\frac{5\pi}{4} + i\sin\frac{5\pi}{4}\right).$$

По теореме Вейля получим

$$M(n) = 2^n - 2^{\frac{n}{2}}\left(\cos\frac{3\pi n}{4} + i\sin\frac{3\pi n}{4}\right) -$$

$$2^{\frac{n}{2}}\left(\cos\frac{5\pi n}{4} + i\sin\frac{5\pi n}{4}\right) \equiv$$

$$2^n + 1 - 2\cdot 2^{\frac{n}{2}}\cos\frac{3\pi n}{4},$$

поскольку

$$\sin\frac{5\pi}{4} = \sin\left(-\frac{3\pi}{4}\right) = -\sin\frac{3\pi}{4}.$$

Для $\cos\frac{3\pi n}{4}$ имеем $\cos\frac{3\pi n}{4} = -\frac{\sqrt{2}}{2}$, если $n = 1 \mod 8$ и $n = 7 \mod 8$ и $\cos\frac{3\pi n}{4} = \frac{\sqrt{2}}{2}$, если $n = 3 \mod 8$ и $n = 5 \mod 8$ (напомним, что n – нечетное).

Окончательно получаем

$$M(n) = 2^n + \sqrt{2^{n+1}} + 1, \text{ если } n = 1 \text{ или } 7 \bmod 8,$$

$$M(n) = 2^n - \sqrt{2^{n+1}} + 1, \text{ если } n = 3 \text{ или } 5 \bmod 8,$$

Для определения порядка группы эллиптической кривой есть ещё CM (complex multiplication) метод, а также алгороитм Скуфа (R.Schoof) и его варианты, однако этот алгоритм, по общепринятому мнению, является довольно громоздким (хотя и полиномиальным). Определив порядок группы эллиптической кривой, мы должны ещё убедиться, что в его разложении на простые множители содержатся большие простые числа (чтобы избежать известных частных случаев вычисления дискретного логарифма). Среди несуперсингулярных кривых для практической имплементации можно брать кривые

$$\mathcal{E}_4 = y^2 + xy = x^3 + x^2 + 1$$

или семейство кривых

$$\mathcal{E}_5 = Y^2 + xy = x^3 + x^2 + \gamma,$$

где $\gamma^3 = \gamma + 1,\ \gamma \in GF(2^n)$.

Так кривая $\mathcal{E}_4$ над полем $GF(2^{163})$ имеет порядок $2 \cdot P49$, а кривая $\mathcal{E}_5$ над $GF(2^{177})$ имеет порядок $10 \cdot P53$. Отметим также, что кривая $\mathcal{E}_4$ имеет порядки с большим простым числом и для случаев $n = 283, 311, 331, 347$ и 359. Фрагмент таблицы для пятичленов степени 163 приведен в приложении.

3.3 О реализации алгоритмов

3.3.1 Библиотека программных функций

На основе предложенных в настоящем учебном пособии алгоритмов на языке программирования C^{++} была реализована библиотека ECCMSUMPEI эффективных программ для арифметики эллиптических кривых в конечных полях. Эти алгоритмы могут составить основу для многих программ коммуникаций, включающих основные протоколы криптографии с открытым ключом:

передача данных, ключевой обмен, аутотентификация, голосование, электронные платежи и т.д. В библиотеке реализованы функции, относящиеся к арифметике конечных полей: умножение ECC_Product, вычисление обратного многочлена ECC_Invert, возведение в степень ECC_Square. Алгоритм умножения основан на методе Карацубы, использует специфику хранения информации в ЭВМ, а также табличное умножение многочленов небольших степеней. Проведено сравнение разработанного алгоритма умножения с известными, в том числе и недавно изобретенными в течение последних 10-ти лет, алгоритмами умножения. Наши схемы умножения более эффективны. Алгоритмы возведения многочленов в степень и вычисления обратного многочлена основаны на известных оптимальных алгоритмах с использованием заранее заготовленных таблиц и блочного представления коэффициентов многочленов. Оптимальность алгоритмов для арифметики конечных полей позволило разработать эффективные функции вычисления kP по натуральному числу k и точке P на эллиптической кривой. Эта функция реализована в двух вариантах. Одна из них ECC_kP_8 служит для многократного вычисления kP при фиксированном P и вариациях числа k. За счет предварительных вычислений эта функция значительно сокращает время вычислений по сравнению с функцией ECC_kP однократного вычисления kP.

Библиотека реализована под руководством авторов настоящего учебного пособия студентами МЭИ А.В. Ковалевым и О.В. Алексеевой по описанным в нём алгоритмам.

3.3.2 Результаты экспериментов

С целью подтверждения эффективности предложенных в настоящем учебном пособии алгоритмов был проведены компьютерные эксперименты. Анализировалась работа соответствующих программ при реализации операций в поле $GF(2^{173})$, порождаемом неприводимым пятичленом

$$x^{173} + x^8 + x^5 + x^2 + 1.$$

степени 173. В табл. 3 приведены минимальное, усредненное и максимальное время по десяти циклам выполнения указанного в таблице количества операций (Pentium MMX, 233 Мгц) того или

иного вида. Можно видеть, что время выполнения инверсии многочлена примерно на два порядка больше времени умножения и на три порядка больше возведения многочлена в квадрат, что, с одной стороны, подтверждает эффективность использованного алгоритма возведения в квадрат (на порядок быстрее умножения), с другой стороны эффективность алгоритма возведения в степень, поскольку для возведения в степень $2^n - 2, n = 173$, используется не более 1000 умножений. Использование разработанных программ для осуществления арифметических операций в абелевой группе точек эллиптической кривой позволило реализовать важную прикладную операцию умножения точки кривой на скаляр в криптосистеме эллиптических кривых за доли секунды, что резко улучшило быстродействие многих криптографических протоколов, например, протокола распределения ключей для классической криптосистемы.

Данные в таблице получены с использованием экспериментальной библиотеки функций, описанной выше.

Операция	Число операций	Мин	Ср.	Макс
Умножение	10 000	0.170	0.179	0.180
Инвертирование	100	0.12	0.12	0.13
Возведение в квадрат	100 000	0.44	0.44	0.45

Глава 4

Протоколы эллиптической криптографии

4.1 Распределение ключей

4.1.1 Введение

В этой лекции мы рассмотрим два протокола, криптографическая стойкость которых основана на трудности решения проблемы дискретного логарифма. Эта проблема имеет место в каждом случае, когда задана некоторая конечной группа Известна степень $y = g^x$ некоторого её элемента и требуется найти значение показателя степени x – дискретный логарифм элемента y. В одних случаях, например для аддитивной группы, заданной на множестве вычетов множества Z целых чисел по модулю простого числа эта проблема легко решается с использованием алгоритма, подобного алгоритму Евклида, в других случаях, например, для мультипликативной группы на множестве классов вычетов положительных целых чисел по модулю простого числа субполиномиальные алгоритмы для этой проблемы неизвестны. Для группы точек эллиптической кривой сложность проблемы дискретного логарифма не меньше сложности этой проблемы в общей постановке для произвольной группы. Исключение составляют суперсингулярные эллиптические кривые, для некоторых из которых проблема дискретного логарифма решается эффективно.

4.1.2 Распределение ключей для классической криптосистемы (протокол Диффи-Хеллмана)

Использование группы $\mathcal{F}_p^*$

Алиса A и Боб B должны условиться об использовании некоторого большого целого числа в качестве ключа для классической криптосистемы, при этом они должны использовать только открытые каналы: любой посторонний наблюдатель, например Кэтрин (C) может знать сообщения, посылаемые ими друг другу.

Алиса и Боб прежде всего договариваются об использовании простого числа p и о базовом элементе $g \in \mathcal{F}_p^*$. Это предполагается известным Кэтрин, поскольку это согласование осуществляется по открытым каналам. Далее Алиса выбирает случайно некоторое положительное число $k_A < p$ (примерно такое же по порядку, как p), вычисляет наименьший положительный остаток по модулю p от числа g^{k_A} и посылает его Бобу, также по открытому каналу. Аналогично поступает и Боб, в итоге и Алиса имеет число g^{k_B}. Заметим, что выбранные числа k_A и k_B как Алиса, так и Боб сохраняют в секрете, они неизвестны Кэтрин.

Соглашением о ключе для Боба будет число

$$g^{k_A k_B} \in \mathcal{F}_p^* = \{0, 1, 2, \ldots, p-1\},$$

которое вычисляет Боб, возводя в степень k_B своего секретного ключа сообщение g^{k_A}, полученное от Алисы, аналогично соглашением о ключе для Алисы будет число

$$g^{k_B k_A} \in \mathcal{F}_p^*,$$

которое вычисляет Алиса, возводя в степень k_A своего секретного ключа сообщение g^{k_B}, полученное от Боба.

Нетрудно проверить, что полученные по описанному протоколу числа одинаковы, так как

$$g^{k_A{}^{k_B}} = g^{k_B{}^{k_A}} = g^{k_A k_B} = g^{k_B k_A}.$$

Проблема, стоящая перед Кэтрин, имеющей намерение раскрыть секретный ключ, выработанный Алисой и Бобом, называется проблемой Диффи-Хеллмана и заключается в вычислении $g^{k_A k_B}$ по известным g, g^{k_A}, $g^{k_B} \in \mathcal{F}_p^*$. Ясно, что всякий, кто знает,

как эффективно вычислять дискретный логарифм без труда решит и эту проблему. Однако никто не доказал, что эта проблема не может быть решена каким-то другим способом, хотя и существует гипотеза об эквивалентности этих проблем. Итак, предполагается, что секретность протокола Диффи-Хеллмана основана на трудной решаемости проблемы дискретного логарифма.

Использование группы точек эллиптической кривой

Рассмотренный протокол можно адаптировать применительно к группе точек эллиптической кривой следующим образом. Прежде всего заметим, что в качестве ключа классической криптосистемы можно использовать случайную точку (x, y) эллиптической кривой, если Алиса и Боб условятся, как конвертировать ее в натуральное число. Например для этого можно взять одну из её координат, например, x и отобразить ее определенным образом, условившись об использовании некоторого отображения из $\mathcal{F}_q$ во множество натуральных чисел.

Для получения такой секретной точки на двух терминалах открытого канала связи можно использовать следующую модификацию протокола Диффи-Хеллмана.

Допустим, что E – эллиптическая кривая и Q – предварительно согласованная и опубликованная точка этой кривой. Алиса выбирает, сохраняя в секрете, случайное число k_A, вычисляет координаты точки k_AQ и пересылает их Бобу. Аналогично Боб выбирает k_B, вычисляет и пересылает Алисе k_BQ. Общим ключом является точка $P = k_Ak_BQ$. Алиса вычисляет ее умножая на свой секретный ключ k_A сообщение, поступившее от Боба, а Боб вычисляет ее, умножая сообщение, поступившее от Алисы на свой секретный ключ k_B. Ввиду того, что группа точек эллиптической кривой абелева, результат не зависит от порядка вычисления и, следовательно, Алиса и Боб имеют одинаковые точки:

$$k_A(k_BQ) = k_B(k_AQ) = k_Ak_BQ.$$

Теперь Алиса и Боб имеют одинаковые копии искомой секретной точки эллиптической кривой.

Проблема, стоящая перед посторонним наблюдателем, имеющим намерение узнать секретный ключ, заключается в вычисле-

нии $k_A k_B$ по известным Q, $k_A Q$, $k_B Q$, но при неизвестных k_A, k_B. Она называется проблемой Диффи-Хеллмана для эллиптических кривых.

Проблема дискретного логарифма для эллиптических кривых заключается в вычислении числа x, такого, что $P = xQ$, где P и Q - известные точки заданной эллиптической кривой. Ясно, что если эта проблема имеет эффективное решение, то и проблема Диффи-Хеллмана для эллиптических также легко решаема, и рассмотренный протокол не имеет смысла. В то же время гипотеза об эквивалентности проблем дискретного логарифма и проблемы Диффи-Хеллмана для эллиптических кривых не доказана.

4.1.3 Распределение ключей для классической криптосистемы (протокол Massey-Omura)

Пусть E – эллиптическая кривая порядка n, e – целое, $1 < e < n$, взаимно простое с n. Используя алгоритм инвертирования, найдём

$$d = e^{-1} (\mathrm{mod}\ n). \tag{4.1}$$

Будем использовать то обстоятельство, что свойства модульной арифметики над целыми числами и над точками эллиптической кривой идентичны.

Используя e и d из (4.1), и любую точку P эллиптической кривой, можно вычислить

$$Q = eP, \qquad R = dQ.$$

Очевидно, что $Q = P$. Протокол Massey-Omura основан на этой идее, реализуемой с учетом трудной решаемости проблемы определения скалярного множителя, соответствующего данной точке эллиптической кривой относительно базовой точки, умножаемой на этот скаляр, то есть на проблеме дискретного логарифма для эллиптических кривых.

Алиса выбирает число e_A и вычисляет по (1.1) число d_A. Аналогично Боб выбирает e_B и вычисляет d_B. Для каждого из них e – ключ шифрования, а d – ключ дешифрования, поскольку $ed = 1(\mathrm{mod}\ n)$.

Алиса помещает свое сообщение m в некоторую точку P_m эллиптической кривой и, умножая её на свое секретное значение e_A, получает точку

$$p_1 = e_A P_m.$$

Эту точку Алиса посылает Бобу.

Боб вычисляет

$$P_2 = e_B P_1$$

и посылает результат Алисе.

Алиса вычисляет

$$P_3 = d_A P_2$$

и возвращает полученную точку Бобу.

Умножая полученную точку на свой секретный ключ дешифрования, Боб получает точку P_m, соответствующую сообщению m Алисы:

$$P_m = d_B P_3.$$

Действительно, вычисляя P_3, Алиса снимает действие своего ключа шифрования:

$$P_3 = d_A P_2 = d_A(e_B P_1) = d_A(e_B(e_A P_m)) = e_B(d_A(e_A P_m)) = e_B P_m.$$

Следовательно, Боб получает

$$d_B P_3 = d_B(d_A P_m) = P_m.$$

Сообщение m может быть использовано в качестве ключа симметричной криптосистемы. Заметим, что в данном случае не требуется опубликования никакой информации о параметрах протокола, кроме самой эллиптической кривой. Платой за это является необходимость трёхкратной передачи по открытым каналам.

Аналогичный протокол с использованием группы $\mathcal{F}_p^*$ читатель может построить в качестве упражнения.

4.1.4 Передача секретных сообщений по открытым каналам с использованием открытого ключа (модификация Эль-Гамаля)

Использование группы $\mathcal{F}_p^*$

Выбирается простое число p и два случайных числа g и x, оба меньшие, чем p. Вычисляется

$$y = g^x (\mathrm{mod}\ p).$$

Числа y, g, и p публикуются как открытый ключ, x сохранятся как секретный ключ.

Для того, чтобы зашифровать сообщение m выбирается случайное число k, взаимно простое с числом p.

Затем вычисляют

$$a = g^k (\mathrm{mod}\ p) \text{ и}$$

$$b = y^k m (\mathrm{mod}\ p).$$

Пара чисел (a, b) образует криптотекст.

Для расшифрования сообщения вычисляют

$$m = b/a^x (\mathrm{mod}\ p).$$

Поскольку $a^x \equiv g^{kx} (\mathrm{mod}\ p)$ и $b/a^x \equiv y^k m/a^x \equiv g^{xk} m/g : xk \equiv m (\mathrm{mod}\ p) = m$, этот протокол можно использовать для передачи сообщений.

Использование группы точек эллиптической кривой

Рассмотренный выше протокол Диффи-Хеллмана с использованием эллиптических кривых можно приспособить для передачи сообщений по схеме Эль Гамаля. Допустим, что множество сообщений представляется точками эллиптической кривой E ("уложено" в эту кривую условленным способом) и пусть Боб намерен переслать Алисе секретное сообщение $M \in E$. Допустим, Алиса и Боб уже обменялись "половинками" ключа $k_A Q$ и $k_B Q$, как в протоколе Диффи-Хеллмана. Теперь Боб выбирает другое секретное число l, вычисляет и посылает Алисе в качестве криптограммы пару точек эллиптической кривой $(lQ,\ M + l(k_A Q))$.

Для расшифровки сообщения Алиса умножает первую точку пары на свой секретный ключ k_A и затем вычитает результат $k_A lQ = lk_A Q$ из второй точки пары, получая точку M эллиптической кривой, их которой извлекает сообщение m.

4.1.5 Протокол распределения ключей Менезеса-Кью-Ванстона (MQV-протокол)

Рассмотренные выше протоколы обладают тем недостатком, что некоторое третье лицо (Кэтрин) может взять на себя функции посредника в передаче сообщений между двумя абонентами и обладать при этом их секретом.

Действительно, если Алиса и Боб взаимодействуют, например, по протоколу Диффи-Хеллмана, то Кэтрин, перехватив передачу открытого ключа $k_A P$ Алисы, передаст Бобу свой открытый ключ $k_C P$, Боб передаст ей свой открытый ключ $k_B P$, после чего Боб и Кэтрин будут иметь общий закрытый ключ $k_C k_B P$. Далее Если Кэтрин передаст свой открытый ключ также Алисе, то Кэтрин и Алиса будут иметь общий секретный ключ $k_A k_C P$ (он может отличаться от ключа Кэтрин и Боба).

При аккуратных действиях Кэтрин Боб и Алиса не будут знать, что имеется посредник, который, получая сообщение одного абонента, способен его расшифровать и вновь зашифровать с использованием другого закрытого ключа.

Для предотвращения таких действий активного криптоаналитика (его еще называают man_in_between) необходима аутентификация (авторизация) этих кратковременных ключей $k_A P$ и $k_B P$ (ключей одноразового использования), для чего используются публикуемые долговременные ключи $d_A P$ и $d_B P$ (ключи многоразового использования). При этом протокол организуется таким образом, что кратковременный открытый ключ функционально связывается с долговременным и поэтому третье лицо, не сможет стать посредником коммуникаций между двумя абонентами. Более точно, для успеха ему необходимо вмешаться в процессы передачи как долговременных, так и кратковременных ключей и более того воспрепятствовать возможности Алисы и Боба проверить правильность передачи долговременных ключей (на это у

этих абонентов найдется время: долговременные ключи передаются редко).

Использование кратковременного ключа обеспечивает невозможность использования раскрытого при одной из передач секрета для раскрытия секрета, вырабатываемого при последующих передачах.

Математическое обоснование этого протокола проще получается с использованием модульной арифметики целых чисел, хотя имплементация может быть осуществлена с использованием циклического свойства подгруппы точек эллиптической кривой, как будет показано ниже.

В обоих случаях используется двоякая трактовка координат точек эллиптической кривой - 1) как элементов расширения поля, над которым строится кривая, 2) как кодов целых чисел.

В случае использования модульной арифметики над такими числами могут выполняться операции сложения и умножения по модулю n порядка эллиптической кривой, в случае использования арифметики эллиптической кривой на такие числа могут умножаться точки эллиптической кривой (тогда цикличность определяется порядком подгруппы точек эллиптической кривой и знание порядка эллиптической кривой или этой подгруппы для выполнения операций не требуется).

Во всех случаях Алиса и Боб располагают точкой P эллиптической кривой порядка n, над которой и осуществляются все вычисления. Кроме того они знают долговременные и кратковременные ключи друг друга: ключи Боба

$$Q_B = d_B P = (a_B, b_B),$$

$$R_B = k_B P = (x_B, y_B)$$

известны Алисе, а её ключи

$$Q_A = d_A P = (a_A, b_A),$$

$$R_A = k_A P = (x_A, y_A)$$

известны Бобу.

Рассмотрим описание и обоснование протокола с использованием модульной арифметики.

Протоколом предусматриваются три этапа, симметрично выполняемых каждой из сторон.

На первом этапе Алиса и Боб вычисляют числа

$$s_A = k_A + x_A a_A d_A \pmod n,$$
$$(s_B = k_B + x_B a_B d_B \pmod n),$$

(при этом они используют свои секретные данные k_A, d_A и k_B, d_B соответственно.)

На втором этапе они вычисляют точки эллиптической кривой

$$U_A = R_B + x_B a_B Q_B,$$
$$U_B = R_A + x_A a_A Q_A.$$

На третьем этапе Алиса и Боб вычисляют точку эллиптической кривой

$$W = s_A U_A,$$
$$W = s_B U_B.$$

Одно и то же обозначение здесь выбрано не случайно, так как результаты вычислений совпадут:

$$s_A U_A = s_B U_B.$$

Действительно, в соответствии с использованными обозначениями, получим

$$s_A U_A = (k_A + x_A a_A d_A \pmod n)(R_B + x_B a_B Q_B) =$$
$$(k_A + x_A a_A d_A \pmod n)(k_B P + a_B x_B d_B P) =$$
$$(k_A + x_A a_A d_A \pmod n)(k_B + x_B a_B d_B) P =$$
$$(k_A + x_A a_A d_A)(k_B + x_B a_B d_B) \pmod n P.$$

Аналогично получим для Боба:

$$s_B U_B = (k_B + x_B a_B d_B \pmod n)(R_A + x_A a_A Q_A) =$$
$$(k_B + x_B a_B d_B \pmod n)(k_A P + a_A x_A d_A P) =$$
$$(k_B + x_B a_B d_B \pmod n)(k_A + x_A a_A d_A) P =$$
$$(k_B + x_B a_B d_B)(k_A + x_A a_A d_A) \pmod n P.$$

Как видим, в рассмотренной интерпретации протокола модульная числовая арифметика сочетается с арифметикой эллиптической кривой.

Рассмотрим теперь интерпретацию, в которой модульная числовая арифметика не используется и числа s_A, s_B заранее не вычисляются.

Заметим, что Алиса может вычислить точку U_A эллиптической кривой, выполнив умножение точки Q_B на константу a_B, и умножив затем результат на константу x_B и, наконец, сложив полученную точку с точкой R_B. Аналогичным образом Боб может получить точку U_B.

Для получения точки W Алиса и Боб, имея в виду, что надо умножить полученные точки на константы s_A и s_B могут проделать это по следующему алгоритму (описываются действия Алисы):

1) Вычислить $k_A U_A$ (умножая точку эллиптической кривой на константу),

2) Вычислить $x_A(a_A(d_A U_A))$ (последовательно умножая точку U_A на константу d_A, затем результат - на константу a_A и полученную точку - на константу x_A),

3) Сложить две точки эллиптической кривой, полученные в п.п. 1) и 2).

Действия Боба аналогичны.

По окончании исполнения протокола Алиса и Боб располагают секретной точкой W эллиптической кривой, координаты которой могут быть использованы для построения бинарного кода секретного ключа симметричной системы.

4.2 Электронная подпись

4.2.1 Стандарт DSS, алгоритм DSA

В 1991 году правительственный национальный институт стандартов и технологии США утвердил стандарт цифровой подписи *DSS* (Digital Signature Stundard), основанный на специальном алгоритме цифровой подписи *DSA* (Digital Signature Algorithm) для использования в правительственных и коммерческих организаци-

ях. Алгоритм основан на трудности проблемы дискретного логарифма мультипликативной группы поля $\mathcal{F}_p$.

Для инициализации, то есть для подготовки к последующему использованию схемы цифровой подписи каждый пользователь, назовем ее Алиса, должен проделать следующее:

1) выбрать простое число q из примерно 160 бит, для этого используется генератор случайных чисел и тесты простоты;

2) выбрать второе случайное число p, такое, что q является делителем числа $p-1$, и которе состоит из примерно 500 бит (более точно, рекомендуется выбирать число, кратное 64 между 512 и 1024);

3) выбрать образующий элемент α единственной циклической подгруппы группы $\mathcal{F}_p^*$ порядка q (это делается вычислением $\alpha = g^{(p-1)/q} (\mathrm{mod}\ p)$ для случайно выбираемого целого g; если в результате получается число, отличное от единицы, то α является образующим элементом);

4) выбрать случайное целое число a в интервале $0 < a < q$ в качестве секретного ключа и образовать открытый ключ $y = \alpha^a (\mathrm{mod}\ p)$.

Теперь Алиса может подписывать сообщения. Сначала она применяет к открытому тексту m хеш-функцию, получая целое $h(m)$ в интервале $0 < h(m) < q$. Затем она выбирает случайное целое k в том же интервале, вычисляет $r = \alpha^k (\mathrm{mod}\ p)\ (\mathrm{mod}\ q)$ (то есть α^k вычисляется по модулю p и затем результат приводится по модулю меньшего простого числа q). В заключение Алиса находит целое s такое, что $s = k^{-1}\{h(m) + ar\}\ (\mathrm{mod\ q})$. Ее подпись для сообщения m образуется парой чисел (r, s) по модулю q.

Чтобы проверить подпись, Боб, получив аутентифицированный открытый ключ Алисы (p, q, α, y),

а) проверяет, выполняются ли $0 < r < q$ и $0 < s < q$, если нет, то отклоняет подпись,

б) вычисляет $w = s^{-1} \mathrm{mod}\ q$ и $h(m)$,

в) вычисляет $u_1 = s^{-1} h (\mathrm{mod\ q})$ и $u_2 = rw (\mathrm{mod}\ q)$,

г) вычисляет $v = \alpha^{u_1} y^{u_2} (\mathrm{mod}\ p)\ (\mathrm{mod}\ q)$,

д) принимает подпись, если $v = r$, иначе подпись отклоняется.

Достоинством этой схемы является то, что подпись является короткой (два числа примерно по 160 бит каждое). С другой сто-

роны уровень секретности определяется сложностью проблемы дискретного логарифма.

Опишем теперь эллиптический аналог DSA – алгоритм ECDSA.

Порождение ключа для ECDSA. Для простоты будем рассматривать эллиптические кривые над полем $\mathcal{F}_p$, хотя конструкция можно легко адаптировать для эллиптических кривых над другим конечным полем.

Пусть E – эллиптическая кривая, определенная над $\mathcal{F}_p$, и пусть P – точка простого порядка q кривой $E(\mathcal{F}_p)$; эти кривая и точка на ней являются системными параметрами. Напомним, что как и в классическом случае числа q и p, – простые. В отличие от DSA, где q намного меньше p, в $ECDSA$ число q имеет примерно такой же порядок, что и число p. Каждый пользователь, скажем Алиса, выбирает случайное число a в интервале $1 < a < q-1$ и вычисляет $Q = aP$ – свой открытый ключ. Он публикуется при сохранении числа a в качестве секретного ключа Алисы.

ECDSA для вычисления цифровой подписи. Чтобы подписать своё сообщение, Алиса проделывает следующее:

1)Выбирает случайное целое k в интервале $1 < k < q-1$.

2)Вычисляет $kP = (x_1, y_1)$ и $r = x_1 (\mathrm{mod}\ q)$ (то есть x_1 получается из целого числа между 0 и $p-1$ приведением по модулю q). Если $r = 0$ то возврат к п. 1). (Если $r = 0$, то уравнение подписи $s = k^{-1}(h(m) + ar)(\mathrm{mod}\ q)$ не зависит от секретного ключа a; следовательно, 0 не подходит в качестве значения для a.)

3)Вычисляет $k^{-1} (\mathrm{mod}\ q)$.

4) Вычисляет $s = k^{-1}(h(m)+xr)(\mathrm{mod} q)$, где $h(m)$ – хэш-значение сообщения m. Если $s = 0$, то возвращение к п.1); (Если $s = 0$, то значение $s^{-1}\ (\mathrm{mod}\ q)$, требуемое на шаге 3) верификации подписи, не существует. Заметим, что если k выбирается случайно, то вероятность того, что либо $r = 0$, либо $s = 0$, исчезающе мала.)

5) Подписью для сообщения m является пара чисел (r, s).

ECDSA для верификации подписи. Для проверки (верификации) подписи Алисы Боб должен выполнить следующие действия:

1) Получить авторизированную копию открытого ключа Q Алисы.

2) Проверить, что r и s являются целыми числами в интервале

$[1, q-1]$.

3) Вычислить $w = s^{-1} (\mathrm{mod}\ q)$ и $h(m)$.

4) Вычислить $u_1 = h(m)w (\mathrm{mod}\ q)$ и $u_2 = \mathrm{к}w (\mathrm{mod}\ q)$.

5)Вычислить $(x_0, y_0) = u_1 P + u_2 Q$ и $v = x_0 (\mathrm{mod}\ q)$.

6) Принять подпись только в том случае, если $v = r$.

Основное различие между ECDSA и DSA состоит в способе образования r. DSA вычисляет r как случайную степень $\alpha^k (\mathrm{mod}\, p)$, приведенную по модулю q, получая тем самым целое в интервале $[1, q-1]$. (Напомним, что в DSA q является 160-битовым делителем числа $p-1$ и α является элементом порядка q группы $\mathcal{F}_p^*$.) ECDSA формирует целое r взятием x-координаты точки kP и приведением ее по модулю q.

Для получения того же уровня секретности, что и при применении DSA, параметр q должен быть 160-битовым. В этом случае DSA и ECDSA имеют одинаковую длину в битах (320 бит).

Вместо использования E и P в качестве глобальных системных параметров, можно фиксировать только поле $\mathcal{F}_p$ для всех пользователей и позволить каждому пользователю выбирать свою собственную эллиптическую кривую E и точку $P \in E(\mathcal{F}_p)$. В этом случае определенное уравнение кривой E, координаты точки P, а также порядок q этой точки P должны быть включены в открытый ключ пользователя. Если поле $\mathcal{F}_p$ фиксировано, то аппаратная и программные составляющие могут быть построены так, чтобы оптимизировать вычисления в этом поле. В то же время имеется огромное количество вариантов выбора эллиптической кривой над полем $\mathcal{F}_p$.

4.2.2 Обобщенная схема электронной подписи Эль Гамаля

Обобщенная схема электронной подписи Эль Гамаля работает в любой абелевой группе.

Для работы по этой схеме каждый участник

1. Выбирает подходящую циклическую группу G, порядка n её образующий элемент α (ниже используется мультипликативное представление группы).

2. Выбирает случайное целое число a, $1 \leq a \leq n-1$ и вычисляет элемент $y = \alpha^a$.

3. Открытым ключом Алисы является пара (α, y) и описание операции умножения ее группы, её секретный ключ есть a.

Опишем как Алиса подписывает документ и как Боб проверяет подпись.

Для формирования подписи, сопровождающей документ m, Алиса должна выполнить следующие действия

а) Выбрать случайное секретное целое k, $1 \leq k \leq n-1$, взаимно простое с n : $(k, n) = 1$.

б) Вычислить элемент $r = \alpha^k$ группы.

в) Вычислить $k^{-1}(\mathrm{mod}\ n)$.

г) Вычислить $h(m)$ и $h(r)$, где h – используемая хеш-функция.

д) Вычисляет $s = k^{-1}\{h(m) - ah(r)\}(\mathrm{mod}\ n)$.

е) Цифровой подписью является пара (r, s).

Для проверки цифровой подписи Алисы на документе m Боб должен проделать следующие действия:

а) Получить авторизированную версию открытого ключа Алисы (α, y).

б) Вычислить $h(m)$ и $h(r)$.

в) Вычислить $v_1 = y^{h(r)} \cdot r^s$.

г) Вычислить $v_2 = \alpha^{h(m)}$.

д) Принять подпись, если $v_1 = v_2$, и отклонить её в противном случае.

Заметим, что генерация подписи требует вычислений как в группе G, так и в группе Z_n, в же время проверка подписи связана с вычислениями только в группе G.

Рассмотренный алгоритм наиболее удачно может быть реализован при использовании в качестве группы G группы точек эллиптической кривой над конечным полем $\mathcal{F}_q$. Проблема дискретного логарифма в этой группе гораздо сложнее, чем в мультипликативной группе конечного поля $\mathcal{F}_q$. Отсюда следует, что может быть выбрано меньшее q, чем в случае имплементации в группе $\mathcal{F}_q^*$.

4.2.3 Схема электронной подписи Эль Гамаля с возвратом сообщения (Nyberg-Rueppel алгоритм)

Во всех рассмотренных выше схемах электронной подписи для верификации подписи требовалось само сообщение. Приводимая

ниже схема не использует текст сообщения при верификации подписи, а напротив, выдает подписанное сообщение при положительном результате верификации. При этом сообщение перед генерацией подписи должно быть преобразовано так, чтобы в него была включена избыточная информация.

Пространством сообщений здесь является $\mathcal{M} = Z_p^*$, где p – простое, в качестве пространства $\mathcal{M}_S$ подготовленных к подписанию сообщений используется декартово произведение $M_S = Z_p \times Z_q$, где q – простое, являющееся делителем $(p-1)$. Пусть R – избыточная функция (инъекция) из пространства сообщений $\mathcal{M}$ в пространство подписанных сообщений $\mathcal{M}_S$. Например, значение $R(m)$ может быть $(m, m(\mathrm{mod}\ q)$. Через R^{-1} обозначается функция из образа функции R в её область определения: $R^{-1} : Im(R) \to \mathcal{M}$, например, $R^{-1}(m, m(\mathrm{mod}\ q) = m$.

Алгоритм генерации ключа тот же, что и в алгоритме DSA, с тем отличием, что на различие размеров p и q не накладывается ограничений (то есть q может быть того же порядка, что и p).

Алгоритм генерации подписи на сообщении m (действия Алисы) следующий:

а) Вычислить $\tilde{m} = R(m)$,

б) Выбрать случайное секретное целое число $k, 1 \le k \le q-1$, и вычислить $r = \alpha^{-k}\ (\mathrm{mod} p)$.

в) Вычислить $e = \tilde{m}r\ (\mathrm{mod}\ p)$.

г) Вычислить $s = (ae + k)\ (\mathrm{mod}\ q)$.

д) Объявить пару (e, s) как подпись на сообщении m.

Алгоритм верификации подписи (действия Боба) следующий:

а) Получить авторизированную копию открытого ключа Алисы.

б) Проверить, что $0 < e < p$ и $0 < s < p$, если нет, то отклонить подпись.

г) Вычислить $v = \alpha^s y^{-e}\ (\mathrm{mod}\ p)$ и $\tilde{m} = ve\ (\mathrm{mod}\ p)$.

д) Проверить, что $\tilde{m} \in \mathcal{M}_R$; если нет, то отклонить подпись.

е) Извлечь (восстановить) сообщение $m = R^{-1}(\tilde{m})$.

Доказательство корректности алгоритма весьма коротко:

По алгоритму генерации подписи

$$v \equiv \alpha^s y^{-e} \equiv \alpha^{s-ae} \equiv \alpha^k (\mathrm{mod\ p}).$$

Таким образом,

$$ve \equiv \alpha^k \tilde{m} \alpha^{-k} \equiv \tilde{m} (\mathrm{mod\ p}),$$

что и требуется.

4.2.4 Схема Nyberg-Rueppel электронной подписи с использованием группы точек эллиптической кривой

Пусть $e = h(m)$ - значение хеш-функции h для документа m. пусть E точка из группы точек эллиптической кривой, P – базовая точка открытого ключа, n – порядок этой точки, s – секретный ключ подписывающего документ участника. Открытым ключом последнего является точка

$$Q = sP.$$

Предполагается, что порядок точки P равен n.

Алгоритм генерации подписи следующий:

1) Образовать случайную битовую строку k и вычислить

$$R = kP.$$

2) Используя x-компоненту точки R как целое число, вычислить

$$c = x + e \ (\mathrm{mod}\ n), \tag{4.2}$$

$$d = k - sc (\mathrm{mod}\ n). \tag{4.3}$$

Пара (c, d) является подписью для документа m, такого, что

$$h(m) = e.$$

Для проверки, что $h(m)$ является корректным хеш-значением, выполняется следующий алгоритм:

1) Вычислить

$$R' = dP + cQ. \tag{4.4}$$

2) используя x-компоненту R', вычислить

$$e' = c - x'(\ \mathrm{mod}\ n). \tag{4.5}$$

3) если полученное значение совпадает с хеш-значением $h(m)$, то последнее удостоверяется.

Рассмотрим эту схему более подробно.

Точка P умножается на случайное число k, и получается точка R, x-компонента этой точки R также является случайным числом. Прибавление этой точки к хеш-значению $e = h(m)$ по модулю n (порядка точки P) эффективно маскирует это хеш-значение.

Этап верификации позволяет восстановить это замаскированное хеш-значение, не допуская никакой утечки информации, которая позволила бы активному криптоаналитику изменить документ и хеш-значение так, чтобы измененный документ воспринимался бы как корректный.

При вычислении R соединяются случайные данные с секретным ключом, так же с использованием модульной арифметики. Поскольку модуль равен порядку точки эллиптической кривой, то же самое можно делать с точками эллиптической кривой.

Действительно, вместо вычисления c по (4.2) вычислим

$$cP = xP + eP \tag{4.6}$$

над кривой E. Далее, вместо (4.3) можно использовать уравнение

$$dP = kP - s(cP). \tag{4.7}$$

Теперь можно использовать выражение (4.2), подставляя в него вычисленные точки cP и dP. При этом (4.4) умножается на s, секретный ключ подписывающего сообщение участника. Но публично известна только точка Q, замаскированная версия секретного ключа.

Это означает, что после прибавления к (4.5) второго терма из выражения (4.2), а именно

$$cQ = c(sP),$$

получается (возвращается) исходная точка R эллиптической кривой.

Если теперь x-компоненту этой точки вычесть из значения c, входящего в подпись, то восстановится хеш-значение $e = h(m)$. Если это восстановленное значение совпадает с хеш-значением $h(m')$, вычисленным по полученному сообщению m', то можно считать, что последнее мог подписать только обладатель секретного

ключа s и что ни сообщение, ни его хеш-значение не было изменено активным криптоаналитиком или вследствие ошибок при передаче или хранении.

Список литературы

[1] Salomaa A. Public Key Cryptograhy. Springer-Verlag, 1990 (second edition – Springer-Verlag, 1997). Русский перевод: Саломаа А. Криптография с открытым ключом. М., Мир, 1996.

[2] Алексеева О.В., Болотов А.А., Гашков С.Б., Лиссук М. О методах вычисления кратных для точек эллиптических кривых// Вестник МЭИ, вып.4. 2000 г.

[3] Болотов А.А., Гашков С.Б., Хохлов Р.А. О сложности алгоритмов построения неприводимых трехчленов и пятичленов над конечными полями// Интеллектуальные системы, т.4. вып.3-4, 1999.

[4] Болотов А.А., Гашков С.Б., Фролов А.Б., Часовских А.Б. О методах реализации умножения многочленов над конечными полями// Вестник МЭИ, вып.3, 2000.

[5] Болотов А.А., Гашков С.Б., Фролов А.Б., Часовских А.А. О методах имплементации арифметических операций в конечных полях// Вестник МЭИ, вып.4, 2000.

[6] Болотов А.А., Гашков С.Б., Фролов А.Б., Часовских А.Б. Программные и схемные методы умножения многочленов для эллиптической криптографии// Известия РАН. Теория и системы управления, N5, 2000.

[7] Болотов А.А, Гашков С.Б., Фролов А.Б., Часовских А.А. О методах имплементации арифметических операций в криптографических системах// Известия РАН. Теория и системы управления, N6, 2000.

[8] Карацуба А.А., Офман Ю.П., Умножение многозначных чисел на автома- тах// ДАН СССР, т.145, N2, 1962

[9] Берлекемп Е. Алгебраическая теория кодирования. М., Мир, 1971.

[10] Гашков С.Б., Чубариков В.Н. Арифметика. Алгоритмы. Сложность вычислений. М., Наука, 1996 (2-е изд. – М., Высш. шк., 2000) .

[11] Knut D. The Art of computer programming, v.2. Addison-Wesley Publ., 1981. Русский перевод: Кнут Д. Искусство программирования на ЭВМ. т.2. М., Мир, 1976.

[12] Лупанов О.Б. Асимптотические оценки сложности управляющих систем. М., Изд-во МГУ, 1987.

[13] Koblitz N. Algebraic Aspects of Cryptography. Springer-Verlag, 1998.

[14] Menezes A., van Oorschot P., Vanstone S. Handbook of Applied Cryptography. CRC Press, 1997.

[15] Menezes A. Elliptic Curve Public Key Cryptosystems. Kluwer, 1993.

[16] Rosing M. Implementing Elliptic Curve Cryptography. Manning, 1999.

Приложение А

Таблицы неприводимых трехчленов

А.1 Неприводимые трехчлены степени 151- 175

$1+x^{3}+x^{151}$	$1+x^{8}+x^{153}$	$1+x^{40}+x^{159}$	$1+x^{132}+x^{167}$
$1+x^{9}+x^{151}$	$1+x^{145}+x^{153}$	$1+x^{119}+x^{159}$	$1+x^{161}+x^{167}$
$1+x^{15}+x^{151}$	$1+x^{152}+x^{153}$	$1+x^{125}+x^{159}$	$1+x^{34}+x^{169}$
$1+x^{31}+x^{151}$	$1+x^{15}+x^{154}$	$1+x^{128}+x^{159}$	$1+x^{42}+x^{169}$
$1+x^{39}+x^{151}$	$1+x^{139}+x^{154}$	$1+x^{5}+x^{160}$	$1+x^{57}+x^{169}$
$1+x^{43}+x^{151}$	$1+x^{62}+x^{155}$	$1+x^{155}+x^{160}$	$1+x^{84}+x^{169}$
$1+x^{46}+x^{151}$	$1+x^{93}+x^{155}$	$1+x^{18}+x^{161}$	$1+x^{85}+x^{169}$
$1+x^{51}+x^{151}$	$1+x^{9}+x^{156}$	$1+x^{39}+x^{161}$	$1+x^{112}+x^{169}$
$1+x^{63}+x^{151}$	$1+x^{11}+x^{156}$	$1+x^{60}+x^{161}$	$1+x^{127}+x^{169}$
$1+x^{66}+x^{151}$	$1+x^{21}+x^{156}$	$1+x^{101}+x^{161}$	$1+x^{135}+x^{169}$
$1+x^{67}+x^{151}$	$1+x^{39}+x^{156}$	$1+x^{122}+x^{161}$	$1+x^{11}+x^{170}$
$1+x^{70}+x^{151}$	$1+x^{57}+x^{156}$	$1+x^{143}+x^{161}$	$1+x^{23}+x^{170}$
$1+x^{81}+x^{151}$	$1+x^{61}+x^{156}$	$1+x^{27}+x^{162}$	$1+x^{147}+x^{170}$
$1+x^{84}+x^{151}$	$1+x^{63}+x^{156}$	$1+x^{63}+x^{162}$	$1+x^{159}+x^{170}$
$1+x^{85}+x^{151}$	$1+x^{65}+x^{156}$	$1+x^{81}+x^{162}$	$1+x^{1}+x^{172}$
$1+x^{88}+x^{151}$	$1+x^{91}+x^{156}$	$1+x^{99}+x^{162}$	$1+x^{7}+x^{172}$
$1+x^{100}+x^{151}$	$1+x^{93}+x^{156}$	$1+x^{135}+x^{162}$	$1+x^{81}+x^{172}$
$1+x^{105}+x^{151}$	$1+x^{95}+x^{156}$	$1+x^{37}+x^{166}$	$1+x^{91}+x^{172}$
$1+x^{108}+x^{151}$	$1+x^{99}+x^{156}$	$1+x^{129}+x^{166}$	$1+x^{165}+x^{172}$
$1+x^{112}+x^{151}$	$1+x^{117}+x^{156}$	$1+x^{6}+x^{167}$	$1+x^{171}+x^{172}$
$1+x^{120}+x^{151}$	$1+x^{135}+x^{156}$	$1+x^{35}+x^{167}$	$1+x^{13}+x^{174}$
$1+x^{136}+x^{151}$	$1+x^{145}+x^{156}$	$1+x^{59}+x^{167}$	$1+x^{57}+x^{174}$
$1+x^{142}+x^{151}$	$1+x^{147}+x^{156}$	$1+x^{77}+x^{167}$	$1+x^{117}+x^{174}$
$1+x^{148}+x^{151}$	$1+x^{31}+x^{159}$	$1+x^{90}+x^{167}$	$1+x^{161}+x^{174}$
$1+x^{1}+x^{153}$	$1+x^{34}+x^{159}$	$1+x^{108}+x^{167}$	$1+x^{6}+x^{175}$

А.2 Неприводимые трехчлены степени 2000-2100

$1 + x^{169} + x^{2001}$
$1 + x^{475} + x^{2001}$
$1 + x^{511} + x^{2001}$
$1 + x^{752} + x^{2001}$
$1 + x^{860} + x^{2001}$
$1 + x^{441} + x^{2004}$
$1 + x^{533} + x^{2004}$
$1 + x^{917} + x^{2006}$
$1 + x^{205} + x^{2007}$
$1 + x^{314} + x^{2007}$
$1 + x^{427} + x^{2007}$
$1 + x^{523} + x^{2007}$
$1 + x^{737} + x^{2007}$
$1 + x^{859} + x^{2007}$
$1 + x^{889} + x^{2007}$
$1 + x^{54} + x^{2009}$
$1 + x^{150} + x^{2009}$
$1 + x^{710} + x^{2009}$
$1 + x^{771} + x^{2009}$
$1 + x^{849} + x^{2009}$
$1 + x^{459} + x^{2010}$
$1 + x^{819} + x^{2010}$
$1 + x^{42} + x^{2015}$
$1 + x^{344} + x^{2015}$
$1 + x^{558} + x^{2015}$
$1 + x^{714} + x^{2015}$
$1 + x^{992} + x^{2015}$
$1 + x^{330} + x^{2017}$
$1 + x^{540} + x^{2017}$
$1 + x^{589} + x^{2017}$
$1 + x^{81} + x^{2020}$
$1 + x^{325} + x^{2020}$
$1 + x^{543} + x^{2020}$
$1 + x^{945} + x^{2020}$
$1 + x^{349} + x^{2022}$
$1 + x^{409} + x^{2022}$
$1 + x^{165} + x^{2023}$
$1 + x^{430} + x^{2023}$
$1 + x^{549} + x^{2023}$
$1 + x^{751} + x^{2023}$
$1 + x^{274} + x^{2025}$
$1 + x^{289} + x^{2025}$
$1 + x^{859} + x^{2025}$
$1 + x^{1001} + x^{2025}$
$1 + x^{1003} + x^{2025}$
$1 + x^{475} + x^{2026}$
$1 + x^{93} + x^{2028}$
$1 + x^{301} + x^{2028}$
$1 + x^{723} + x^{2028}$
$1 + x^{793} + x^{2028}$
$1 + x^{831} + x^{2028}$
$1 + x^{845} + x^{2028}$
$1 + x^{847} + x^{2028}$
$1 + x^{881} + x^{2028}$
$1 + x^{386} + x^{2031}$
$1 + x^{400} + x^{2031}$
$1 + x^{881} + x^{2033}$
$1 + x^{909} + x^{2033}$
$1 + x^{143} + x^{2034}$
$1 + x^{71} + x^{2036}$
$1 + x^{195} + x^{2036}$
$1 + x^{783} + x^{2036}$
$1 + x^{857} + x^{2036}$
$1 + x^{155} + x^{2039}$
$1 + x^{651} + x^{2039}$
$1 + x^{840} + x^{2039}$
$1 + x^{947} + x^{2039}$
$1 + x^{735} + x^{2041}$
$1 + x^{771} + x^{2041}$
$1 + x^{45} + x^{2044}$
$1 + x^{85} + x^{2044}$
$1 + x^{289} + x^{2044}$
$1 + x^{639} + x^{2044}$
$1 + x^{655} + x^{2044}$
$1 + x^{789} + x^{2044}$
$1 + x^{3} + x^{2047}$
$1 + x^{66} + x^{2047}$
$1 + x^{165} + x^{2047}$
$1 + x^{411} + x^{2047}$
$1 + x^{495} + x^{2047}$
$1 + x^{511} + x^{2047}$
$1 + x^{817} + x^{2047}$
$1 + x^{124} + x^{2049}$
$1 + x^{140} + x^{2049}$
$1 + x^{433} + x^{2049}$
$1 + x^{523} + x^{2049}$
$1 + x^{934} + x^{2049}$
$1 + x^{323} + x^{2052}$
$1 + x^{589} + x^{2052}$
$1 + x^{627} + x^{2052}$
$1 + x^{201} + x^{2054}$
$1 + x^{497} + x^{2054}$
$1 + x^{11} + x^{2055}$
$1 + x^{392} + x^{2055}$
$1 + x^{245} + x^{2057}$
$1 + x^{633} + x^{2057}$
$1 + x^{343} + x^{2058}$
$1 + x^{427} + x^{2058}$
$1 + x^{591} + x^{2058}$
$1 + x^{667} + x^{2058}$
$1 + x^{387} + x^{2060}$
$1 + x^{735} + x^{2060}$
$1 + x^{981} + x^{2060}$
$1 + x^{987} + x^{2060}$
$1 + x^{48} + x^{2063}$
$1 + x^{353} + x^{2063}$
$1 + x^{570} + x^{2063}$
$1 + x^{674} + x^{2063}$
$1 + x^{719} + x^{2063}$
$1 + x^{770} + x^{2063}$
$1 + x^{97} + x^{2065}$
$1 + x^{382} + x^{2065}$
$1 + x^{918} + x^{2065}$
$1 + x^{71} + x^{2066}$
$1 + x^{237} + x^{2070}$
$1 + x^{253} + x^{2073}$
$1 + x^{557} + x^{2073}$
$1 + x^{231} + x^{2074}$
$1 + x^{583} + x^{2074}$
$1 + x^{851} + x^{2076}$
$1 + x^{897} + x^{2076}$
$1 + x^{933} + x^{2076}$
$1 + x^{35} + x^{2079}$
$1 + x^{134} + x^{2079}$
$1 + x^{283} + x^{2079}$
$1 + x^{581} + x^{2079}$
$1 + x^{590} + x^{2079}$
$1 + x^{721} + x^{2079}$
$1 + x^{755} + x^{2079}$
$1 + x^{796} + x^{2079}$
$1 + x^{854} + x^{2079}$
$1 + x^{928} + x^{2079}$
$1 + x^{959} + x^{2079}$
$1 + x^{998} + x^{2079}$
$1 + x^{467} + x^{2081}$
$1 + x^{662} + x^{2081}$
$1 + x^{980} + x^{2081}$
$1 + x^{523} + x^{2082}$
$1 + x^{261} + x^{2086}$
$1 + x^{673} + x^{2086}$
$1 + x^{141} + x^{2087}$
$1 + x^{225} + x^{2087}$
$1 + x^{569} + x^{2087}$
$1 + x^{737} + x^{2087}$
$1 + x^{150} + x^{2089}$
$1 + x^{349} + x^{2089}$
$1 + x^{357} + x^{2089}$
$1 + x^{846} + x^{2089}$
$1 + x^{853} + x^{2089}$
$1 + x^{909} + x^{2089}$
$1 + x^{645} + x^{2094}$
$1 + x^{933} + x^{2094}$
$1 + x^{256} + x^{2095}$
$1 + x^{457} + x^{2095}$
$1 + x^{607} + x^{2095}$
$1 + x^{691} + x^{2095}$
$1 + x^{119} + x^{2097}$
$1 + x^{19} + x^{2098}$
$1 + x^{35} + x^{2100}$
$1 + x^{49} + x^{2100}$
$1 + x^{61} + x^{2100}$
$1 + x^{193} + x^{2100}$
$1 + x^{225} + x^{2100}$
$1 + x^{325} + x^{2100}$
$1 + x^{385} + x^{2100}$
$1 + x^{435} + x^{2100}$
$1 + x^{511} + x^{2100}$
$1 + x^{583} + x^{2100}$
$1 + x^{635} + x^{2100}$
$1 + x^{637} + x^{2100}$
$1 + x^{675} + x^{2100}$
$1 + x^{923} + x^{2100}$
$1 + x^{975} + x^{2100}$
$1 + x^{1009} + x^{2100}$

Приложение В

Фрагменты таблиц неприводимых пятичленов

В.1 Неприводимые пятичлены степени 163

$1 + x^{3} + x^{6} + x^{7} + x^{163}$
$1 + x^{4} + x^{58} + x^{79} + x^{163}$
$1 + x^{2} + x^{27} + x^{83} + x^{163}$
$1 + x^{42} + x^{51} + x^{85} + x^{163}$
$1 + x^{2} + x^{82} + x^{86} + x^{163}$
$1 + x^{15} + x^{54} + x^{88} + x^{163}$
$1 + x^{10} + x^{41} + x^{89} + x^{163}$
$1 + x^{7} + x^{68} + x^{90} + x^{163}$
$1 + x^{8} + x^{22} + x^{92} + x^{163}$
$1 + x^{86} + x^{91} + x^{92} + x^{163}$
$1 + x^{2} + x^{11} + x^{93} + x^{163}$
$1 + x^{25} + x^{95} + x^{96} + x^{163}$
$1 + x^{2} + x^{8} + x^{97} + x^{163}$
$1 + x^{17} + x^{97} + x^{98} + x^{163}$
$1 + x^{6} + x^{10} + x^{99} + x^{163}$
$1 + x^{6} + x^{18} + x^{99} + x^{163}$
$1 + x^{8} + x^{26} + x^{99} + x^{163}$
$1 + x^{6} + x^{58} + x^{99} + x^{163}$
$1 + x^{14} + x^{77} + x^{100} + x^{163}$
$1 + x^{21} + x^{77} + x^{100} + x^{163}$
$1 + x^{23} + x^{77} + x^{102} + x^{163}$
$1 + x^{24} + x^{49} + x^{104} + x^{163}$
$1 + x^{101} + x^{114} + x^{124} + x^{163}$
$1 + x^{94} + x^{135} + x^{136} + x^{163}$

$1 + x^{2} + x^{9} + x^{137} + x^{163}$
$1 + x^{115} + x^{137} + x^{143} + x^{163}$
$1 + x^{117} + x^{137} + x^{143} + x^{163}$
$1 + x^{81} + x^{85} + x^{144} + x^{163}$
$1 + x^{25} + x^{86} + x^{144} + x^{163}$
$1 + x^{10} + x^{87} + x^{144} + x^{163}$
$1 + x^{2} + x^{38} + x^{145} + x^{163}$
$1 + x^{26} + x^{54} + x^{145} + x^{163}$
$1 + x^{9} + x^{65} + x^{145} + x^{163}$
$1 + x^{17} + x^{65} + x^{145} + x^{163}$
$1 + x^{5} + x^{58} + x^{153} + x^{163}$
$1 + x^{33} + x^{58} + x^{153} + x^{163}$
$1 + x^{119} + x^{154} + x^{156} + x^{163}$
$1 + x^{104} + x^{114} + x^{157} + x^{163}$
$1 + x^{135} + x^{146} + x^{157} + x^{163}$
$1 + x^{4} + x^{63} + x^{158} + x^{163}$
$1 + x^{21} + x^{157} + x^{162} + x^{163}$
$1 + x^{56} + x^{157} + x^{162} + x^{163}$
$1 + x^{81} + x^{157} + x^{162} + x^{163}$
$1 + x^{17} + x^{158} + x^{162} + x^{163}$
$1 + x^{23} + x^{158} + x^{162} + x^{163}$
$1 + x^{12} + x^{159} + x^{162} + x^{163}$
$1 + x^{77} + x^{161} + x^{162} + x^{163}$
$1 + x^{155} + x^{161} + x^{162} + x^{163}$

B.2 Неприводимые пятичлены степени 173

$1 + x^{2} + x^{5} + x^{8} + x^{173}$
$1 + x^{5} + x^{7} + x^{10} + x^{173}$
$1 + x^{10} + x^{11} + x^{16} + x^{173}$
$1 + x^{15} + x^{16} + x^{17} + x^{173}$
$1 + x^{20} + x^{21} + x^{23} + x^{173}$
$1 + x^{20} + x^{23} + x^{29} + x^{173}$
$1 + x^{25} + x^{26} + x^{30} + x^{173}$
$1 + x^{30} + x^{31} + x^{34} + x^{173}$
$1 + x^{30} + x^{33} + x^{35} + x^{173}$
$1 + x^{32} + x^{36} + x^{38} + x^{173}$
$1 + x^{32} + x^{38} + x^{41} + x^{173}$
$1 + x^{36} + x^{38} + x^{44} + x^{173}$
$1 + x^{36} + x^{42} + x^{45} + x^{173}$
$1 + x^{41} + x^{42} + x^{46} + x^{173}$
$1 + x^{43} + x^{47} + x^{48} + x^{173}$
$1 + x^{48} + x^{50} + x^{51} + x^{173}$
$1 + x^{51} + x^{52} + x^{55} + x^{173}$
$1 + x^{51} + x^{56} + x^{63} + x^{173}$
$1 + x^{55} + x^{59} + x^{66} + x^{173}$
$1 + x^{55} + x^{59} + x^{67} + x^{173}$
$1 + x^{63} + x^{68} + x^{69} + x^{173}$
$1 + x^{63} + x^{70} + x^{73} + x^{173}$
$1 + x^{67} + x^{74} + x^{75} + x^{173}$
$1 + x^{72} + x^{75} + x^{77} + x^{173}$
$1 + x^{74} + x^{75} + x^{79} + x^{173}$
$1 + x^{76} + x^{79} + x^{80} + x^{173}$
$1 + x^{76} + x^{83} + x^{84} + x^{173}$
$1 + x^{76} + x^{85} + x^{86} + x^{173}$
$1 + x^{83} + x^{87} + x^{89} + x^{173}$
$1 + x^{83} + x^{87} + x^{94} + x^{173}$
$1 + x^{87} + x^{88} + x^{97} + x^{173}$
$1 + x^{87} + x^{89} + x^{98} + x^{173}$
$1 + x^{88} + x^{91} + x^{99} + x^{173}$
$1 + x^{96} + x^{98} + x^{101} + x^{173}$
$1 + x^{98} + x^{99} + x^{106} + x^{173}$
$1 + x^{100} + x^{102} + x^{109} + x^{173}$
$1 + x^{100} + x^{103} + x^{110} + x^{173}$
$1 + x^{106} + x^{109} + x^{111} + x^{173}$
$1 + x^{107} + x^{113} + x^{116} + x^{173}$
$1 + x^{110} + x^{114} + x^{117} + x^{173}$
$1 + x^{110} + x^{117} + x^{122} + x^{173}$
$1 + x^{116} + x^{122} + x^{123} + x^{173}$
$1 + x^{122} + x^{123} + x^{125} + x^{173}$
$1 + x^{125} + x^{126} + x^{130} + x^{173}$
$1 + x^{127} + x^{131} + x^{132} + x^{173}$
$1 + x^{128} + x^{131} + x^{137} + x^{173}$
$1 + x^{132} + x^{135} + x^{141} + x^{173}$
$1 + x^{134} + x^{139} + x^{142} + x^{173}$

Содержание

Анатолий Александрович Болотов
Сергей Борисович Гашков
Александр Борисович Фролов
Анатолий Александрович Часовских

Алгоритмические основы эллиптической криптографии

Москва, 107207, Красноказарменная ул., дом 14,
Московский энергетический институт (технический университет)