О сложности алгоритмов построения неприводимых трехчленов и пятичленов над конечными полями

А.А. Болотов, С.Б. Гашков, Р.А. Хохлов

1. Введение

Зачем нужно искать неприводимые многочлены (в частности, трехчлены и пятичлены) над конечными полями? Знание неприводимых многочленов нужно уже для того, чтобы строить сами эти поля. Поэтому в любой книге по кодированию есть таблицы таких многочленов невысоких степеней. Среди неприводимых многочленов особый интерес представляют примитивные многочлены, то есть такие, корни которых являются примитивными (или порождающими) элементами поля разложения этого многочлена. Примитивные элементы являются основаниями дискретных логарифмов, таблицы которых (и примитивных элементов и логарифмов) также есть в любой книге по кодированию, так как существенно облегчают умножение в конечных полях.

Но представление элементов поля в виде степеней примитивного элемента хотя и облегчает умножение, но сильно затрудняет сложение, поэтому чаще элементы поля представляют в виде векторов, разложенных по стандартному полиномиальному базису, тогда сложение сводится к сложению этих векторов (особенно просто оно выполняется в полях порядка, равного степени двойки, так как сводится к покомпонентной логической операции XOR), а умножение представлет из себя умножение многочленов над исходным конеч-

ным полем коэффициентов, выполняемое по модулю неприводимого многочлена, определяющего рассматриваемое представление поля. Во многих ситуациях, однако, вместо стандартного базиса удобнее так называемый нормальный базис, который можно породить порождающим же элементом стандартного базиса (корнем рассматриваемого неприводимого над данным полем многочлена в своем поле разложения). Но нормальный базис порождается не каждым неприводимым многочленом, и возникает задача тестирования заданного неприводимого многочлена на «нормальность». К счастью, есть явные формулы как для числа неприводимых и примитивных многочленов заданной степени, так и для числа нормальных базисов, которые существуют всегда [7]. Доказано также, что среди нормальных базисов есть даже базисы, чей порождающий элемент является примитивным [7]. Однако во многих случаях важно иметь подобные многочлены с минимальным по возможности числом одночленов, например трехчлены или пятичлены (в полях характеристики два неприводимых «четночленов», очевидно, не бывает). Теоретически такие многочлены предъявить не удается и их приходится строить с помощью компьютеров.

Впрочем, для небольших полей компьютер особенно не нужен. Удивительно, но выдающийся специалист по кодированию и конечным полям Элвин Берлекемп (автор известного алгоритма факторизации многочленов над конечными полями) в своей книге [2] написал в шестидесятые годы, что большие конечные поля представляют только академический интерес. Однако, в те же годы Цирлер и Брилхарт (да и многие другие тоже, см., например, [17],[18],[19],[20]) в работах [21],[22],[23] явно не только с академическим интересом вычисляли с помощью тогдашних суперкомпьютеров таблицы неприводимых и примитивных трехчленов над полем из двух элементов большой размерности. В частности, они привели таблицу всех неприводимых трехчленов над GF(2) степеней от 2 до 1000 включительно. Более новые результаты и таблицы в этом направлении были опубликованы, например, в [24] и [25], где приводятся примеры (первые найденные программой при случайном поиске) примитив-

ных неприводимых трехчленов, а также пятичленов и семичленов, для тех степеней $n < 5000$, для которых известно разложение на множители числа $2^n - 1$. В [26] строятся примитивные неприводимые многочлены над полями F_p для всех простых $p < 100$, лишь бы степень многочлена n удовлетворяла неравенству $p^n < 10^{50}$.

Так что интерес к большим конечным полям, в частности с малой характеристикой, уже давно не академический. Без этих полей немыслима современная криптография с открытым ключом ([30]). В связи с этим важной стала также задача быстрой имплементации арифметики в этих полях, для чего понадобились нормальные базисы (а также самодвойственные, почти самодвойственные и десять лет назад открытые оптимальные нормальные базисы, которых, впрочем, в этой статье мы касаться не будем). Примитивные элементы и многочлены нужны, естественно, и для этого. Отметим попутно, что примитивные многочлены нужны также и для построения линейных рекуррентных последовательностей максимально возможной длины [3], которые применяются, например, при построении датчиков случайных чисел, давно применяемых в классической криптографии.

Раньше подобные темы интересовали у нас только узких специалистов по проблемам передачи и защиты информации, которые о своих этих интересах не распространялись. Сейчас, когда проблемы защиты информации давно вырвались из военно дипломатической плоскости в пространство открытых компьютерных сетей, это уже не узко специфическая тематика, а поле для деятельности, которое интересует многих исследователей и разработчиков. Достаточно отметить, что уже появляются книги (например, [6]), в которых открыто выражается сожаление, что у нас никогда не публиковались (в открытой печати) таблицы неприводимых и примитивных многочленов высоких степеней, а западные таблицы малодоступны.

В предлагаемой статье авторы решили восполнить этот пробел и написать, как самому сделать программу, быстро генерирующую и тестирующую неприводимые многочлены, и проверяющую их на «примитивность» и «нормальность», а заодно и сами написали такую программу на основе собственных вариантов алгоритмов генерации

неприводимых многочленов. В приложениях к статье приведены некоторые фрагменты таблиц, сгенерированных этой программой и дающих представление о ее возможностях. Так например, в приложении Б приведен для иллюстрации полный список (с точностью до возвратных) всех неприводимых трехчленов для степеней n, где $2000 < n \leqslant 2100$, (в продолжение как бы того списка, который встречали авторы в доступной западной литературе и который был только для $n \leqslant 2000$). При предоставлении места авторы готовы опубликовать полные таблицы неприводимых трехчленов и пятичленов высоких степеней, в том числе и таких, которые не встречались в доступной литературе (впрочем, полные таблицы пятичленов уж очень обширны даже при степенях порядка двухсот). Авторы надеются, что после изучения предлагаемой статьи заинтересованные читатели и сами смогут написать программу и порождать с ее помощью необходимые им неприводимые многочлены.

2. Тест на неприводимость многочленов над конечными полями

Для нахождения неприводимого полинома q заданной степени n по заданному простому модулю используется следующий алгоритм, являющийся вариантом известного в теории конечных полей алгоритма (см. например [7], [30]).

Выбирается случайный многочлен q. Если он является трехчленом, то с помощью теоремы Штикельбергера-Суона (см., например, [2]), проверяется, не является ли он приводимым. Указанная теорема позволяет заметить, что трехчлен $x^n + x^k + 1$ над полем $GF(2)$ заведомо будет приводимым

1) при четном n, нечетном k, не равном половине n и $nk/2$ равным 0 или 1 по модулю 4;

2) при четном k и n равном ± 1 по модулю 8 в случае, когда k делит $2n$, и n равным ± 3 по модулю 8 в случае, когда k не делит $2n$.

Кроме того, очевидно, что при четных n и k трехчлен является

квадратом, и поэтому приводим.

Если эта простейшая проверка на приводимость не дала результата (а она результативна с вероятностью асимптотически $1/2$), то 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 выбираем следующий многочлен.

В противном случае выполняем следующий тест, пригодный не только для поля $GF(p)$, но и для произвольного конечного поля, но который на практике быстрее всего работает, когда $p = 2$. Строим 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 $q_{n/s} = x$, то многочлен q приводим, и вычислять последовательность q_k далее индекса n/s не нужно.

Для оценки сложности этого 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 при тестировании «малочленов» эта сложность линейна, а в общем случае не более чем квадратична. Поэтому сложность вычисления последовательности $q_k, k = 0, \ldots, n$ в случае «малочленов» квадратична, а в общем случае не более чем кубична.

2.1. Оценка сложности алгоритма Евклида

Для вычисления НОД многочленов (степени которых не превосходят n), применяя версию алгоритма Евклида, в которой вместо полного деления с остатком применяется вычитание делимого, умноженного на соответствующий одночлен, легко получить оценку сложности в наихудшем случае $O(n^2)$. Отметим, что в среднем этот алгоритм работает существенно быстрее.

Можно использовать теоретически более быстрый вариант Шен-

хаге-Моенка алгоритма Евклида ([14], [16]), который дает в наихудшем случае оценку $O(M(n)\log n)$, где $M(n)$ – сложность умножения многочленов степени n в рассматриваемом поле (изложение этого алгоритма можно найти в [1], [4].) Отметим (см., например, [1]), что сложность вычисления остатка от деления многочлена степени $O(n)$ на многочлен степени n равна $O(M(n))$, а Шенхаге показал [15], что для поля $GF(2)$ справедливо равенство $M(n) = O(n\log n\log\log n)$.

2.2. Оценка сложности теста на неприводимость

Известно ([8]), что количество простых делителей числа n не превосходит по порядку $\frac{\log n}{\log\log n}$), а в среднем оно равно по порядку $\log\log n$. Отсюда вытекает оценка сложности всего теста на неприводимость «малочленов», равная по порядку в худшем случае

$$n^2 + \frac{M(n)\log^2 n}{\log\log n}.$$

Второе слагаемое в ней (оценивающее сложность вычисления НОД) будет асимптотически меньше первого уже при использовании алгоритма Карацубы [10]. В общем случае, согласно сделанному выше замечанию, первое слагаемое надо заменить на $nO(M(n))$, и вся оценка приобретет тот же вид даже при использовании стандартного варианта алгоритма Евклида.

3. Об ускорении поиска неприводимых «малочленов» над конечными полями

Для простоты рассмотрим наиболее важный для практики случай поля $GF(2)$. Для ускорения поиска неприводимых трехчленов x^n+x^k+1 можно заметить, что трехчлены x^n+x^k+1 и $x^n+x^{n-k}+1$ взаимно возвратны друг к другу, то есть получаются друг из друга преобразованием $p(x) \to p^*(x) = x^n p(x^{-1})$, поэтому они будут приводимыми или неприводимыми одновременно, значит достаточно перебрать только трехчлены $x^n + x^k + 1, k < n/2$ (трехчлен $x^n + x^{n/2} + 1$ очевидно является квадратом, то есть приводим). Аналогичным образом можно сократить вдвое перебор и в случае пятичленов.

Как известно, порядки многочленов $p(x)$ и $p^*(x)$ совпадают, поэтому взаимно возвратные многочлены будут примитивными или непримитивными одновременно, поэтому аналогичным образом вдвое сокращается перебор при поиске примитивных многочленов.

Заметим еще, что в случае $(n,k) = t > 1$, где t нечетно, проверку на неприводимость тоже можно ускорить (если t четно, то все такие многочлены очевидно являются квадратами). Для этого заметим, что тогда $p(x) = f(x^t)$, где $f(x)$ – трехчлен степени $m = n/t$, и если он приводим, то и $p(x)$ тоже приводим, поэтому достаточно провести проверку на неприводимость у всех трехчленов степени m, что делается существенно быстрее, чем при прямой проверке.

Если выяснится, что многочлен $f(x)$ неприводим, то неочевидно, что тогда многочлен $p(x) = f(x^t)$ тоже неприводим.

Имеется однако теорема (теорема 3.35 [7]) о том, что если порядок e многочлена $f(x)$ таков, что все простые делители числа t делят e, но не делят $\frac{2^m-1}{e}$, то многочлен $p(x) = f(x^t)$ будет неприводимым порядка et (порядок многочлена $g(x)$ равен r – это по определению означает, что r есть наименьший показатель степени, такой, что $x^r - 1$ делится на $g(x)$). Для проверки указанного условия для e и t разложим t на простые множители,

$$t = p_1^{\alpha_1}\dots p_s^{\alpha_s},$$

что делается со сложностью $o(t)$, определим для каждого p_i максимальный показатель степени β_i, такой что $p_i^{\beta_i}$ делит $2^m - 1$, что делается со сложностью $O(m^2)$, тогда указанное выше условие равносильно тому, что e должно делится на $p_1^{\beta_1}\dots p_s^{\beta_s}$, а так как e – это порядок многочлена $f(x)$, и он не должен быть делителем чисел

$$a_i = \frac{2^m-1}{p_i},$$

значит $x^{a_i} \bmod f(x)$ должно быть отлично от единицы при всех $i =$

$1, \ldots, s$, и это условие равносильно указанному выше, а его проверка выполняется со сложностью $O(sm^2 \log\log m)$, после чего остается (в случае необходимости) со сложностью $O(m^2)$ убедиться в неприводимости многочлена $f(x)$. Заметим, что s не превосходит числа простых делителей у t, то есть не больше $O(\log t/\log\log t)$ (см. [8]), а в среднем значительно меньше. Поэтому $O(sm^2 \log\log m) = o(n^2)$ при $t^2 \log\log t/\log t >> \log\log m$.

Приведенная оценка сложности проверки упомянутого условия слишком грубая, так как в ней оценивается число необходимых умножений в поле $GF(2^m)$ как $O(sm/\log m)$ при m возведениях в квадрат, а сложность каждого умножения согласно оценке Шенхаге как

$$O(m \log m \log\log m).$$

В реальности число умножений может быть не столь большим, так как показатели степени – не произвольные s чисел, а числа вида

$$\frac{2^m - 1}{p_i}.$$

Если, например, m_i – такое наименьшее число, что $2^{m_i} - 1$ кратно p_i, и m_i много меньше m, то, как известно, m кратно m_i и

$$\frac{2^m - 1}{p_i} = b_i \frac{2^m - 1}{2^{m_i} - 1},$$

где

$$b_i = \frac{2^{m_i} - 1}{p_i}.$$

Представляя число

$$\frac{2^m - 1}{2^{m_i} - 1}$$

в виде

$$\frac{a^{m/m_i} - 1}{a - 1},$$

где $a = 2^{m_i}$, можно, предварительно вычислив все степени с маленькими показателями b_i, найти все нужные нам степени, используя только

$$\sum_{i=1}^{s} O(\log m/m_i)$$

умножений, а последняя величина в худшем случае не больше $O(m)$, а в среднем значительно меньше, но число возведений в квадрат при таком методе будет грубо оцениваться как $O(sm)$.

Например, при $n = 1000$ возможные значения t есть только 5 и 25, и для упомянутой проверки достаточно выполнить возведения в $(2^{200} - 1)/5$ и $(2^{40} - 1)/25$ степени. Для этого вычислим $(2^4 - 1)/5 = 3$ степень, и

$$(2^{20} - 1)/25 = 3(1 + 3(1 + 2^4 + 1 + 2^8 + 2^4 + 1 + (2^4 + 1)(2^8 + 1))) =$$

$$= 3(1 + 3(4 + 2^4 + 2^5 + 2^9 + 2^{12})),$$

что требует всего 8 умножений, а потом выполним возведения в $(2^{200} - 1)/(2^4 - 1)5$ и в $2^{20} + 1$ степени. Последняя операция требует лишь одного умножения и 20 кратного возведения в квадрат. Первая операция требует не более 200 возведений в квадрат и не более 12 умножений. Как видим, в реальности общее число операций может быть существенно меньше даже числа n (не считая сложность тестирования многочленов $f(x)$ степени n/t).

Покажем, что если упомянутое выше условие не выполняется, то тестирование проводить не нужно, так как тогда многочлен $f(x^t)$ будет всегда приводимым. Сначала рассмотрим случай, когда $(t, e) =$ 1.

Известно ([7], теоремы 2.47 и 3.5), что произведение всех неприводимых над полем $GF(2)$ многочленов степени m и порядка e и единичными старшими коэффициентами равно Q_e – круговому многочлену порядка e, если m – такое наименьшее число, что $2^m - 1$ кратно e (называемое порядком двойки по модулю e), $e > 1$. Круговой многочлен степени Q_e имеет степень $\phi(e)$, коэффициенты его принадлежат $GF(2)$ и все его корни некратны и лежат в поле $GF(2^m)$,

являясь первообразными корнями e-й степени из единицы, то есть имеют вид $\alpha^s, (s,e) = 1, 1 \leqslant s \leqslant e$, где $\alpha \in GF(2^m), \alpha^e = 1, \alpha^r \neq 1, r = 1, \ldots, e-1$. Отсюда видно, что если $f(x)$ – любой из упомянутых неприводимых многочленов (а их количество, кстати, равно $\phi(e)/m$), и $\alpha^s, (s,e) = 1$ – любой из его корней, лежащих в поле $GF(2^m)$, то $\alpha^r, (r,e) = 1, rt = s \bmod e$, будет корнем многочлена $f(x^t)$ и одновременно корнем какого-то неприводимого над полем $GF(2)$ многочлена $g(x)$ степени m и порядка e (может быть и равного $f(x)$), значит многочлены $g(x)$ и $f(x^t)$ не взаимно просты, а так как $g(x)$ – неприводим, то он должен быть делителем $f(x^t)$, и последний поэтому не является неприводимым, так как его степень равна $mt > m$.

Рассмотрим теперь случай, когда t имеет простой делитель p, не делящий e. Тогда, согласно предыдущему, многочлен $h(x) = f(x^p)$ приводим над полем $GF(2)$, а значит и многочлен $f(x^t) = h(x^{t/p})$ тоже.

Осталось рассмотреть случай, когда все простые делители t делят e и $(t, (2^m-1)/e) > 1$. Тогда $t' = t/(t,(2^m-1)/e) < t$. Если $t' = 1$, то $et = e(t,(2^m-1)/e)$ делит $2^m - 1$, так как $(t,(2^m-1)/e)$ делит $(2^m-1)/e$, значит порядок двойки по модулю et равен $m < mt$, также как и по модулю e. Если же $t' > 1$, пусть p – любой его простой делитель, и $t'' = (t,(2^m-1)/e)p$. Тогда t'' делит $(t,(2^m-1)/e)t' = t$. Проверим, что тогда порядок двойки по модулю et'' будет равен $mp < mt''$.

Действительно, указанный порядок s должен быть кратен m и иметь вид mk, так как иначе $2^s - 1$ не делилось бы на e, но

$$2^{mk} - 1 = \frac{2^{mk}-1}{2^m-1}(1 + 2^m + \ldots + 2^{(k-1)m})$$

будет делится на $et'' = e(t,(2^m-1)/e)p$ тогда и только тогда, когда

$$1 + 2^m + \ldots + 2^{(k-1)m}$$

будет делится на p (ведь $e(t,(2^m-1)/e)$ делит $e(2^m-1)/e = 2^m - 1$, но et'' не делит $2^m - 1$, так как тогда бы число $(t,(2^m-1)/e)p$ делило бы $(2^m-1)/e$ и $t = t'(t,(2^m-1)/e)$ одновременно, что противоречит

тому, что $(t,(2^m-1)/e)$ – наибольший общий делитель чисел t и $(2^m-1)/e)$, а так как $2^m - 1$ делится на e, а значит и на p, то по модулю p указанная сумма будет равна k, и будет кратна p лишь при k кратном p, откуда и следует требуемое. Заметим, что в обоих случаях было доказано существование такого делителя t' у числа t (может быть $t' = t$), что порядок двойки по модулю et' будет меньше mt'.

Уже отмечалось выше, что произвольный неприводимый над полем $GF(2)$ многочлен $f(x)$ степени m и порядка e является делителем кругового многочлена Q_e, а значит многочлен $f(x^{t'})$ делит многочлен $Q_e(x^{t'})$, а так как все простые делители t делят e, то, как показано в доказательстве теоремы 3.35 в [7], из свойств круговых многочленов вытекает, что $Q_e(x^{t'}) = Q_{et'}(x)$, значит многочлен $f(x^{t'})$ делит круговой многочлен $Q_{et'}(x)$, но как показано там же, согласно теореме 2.47 степень каждого неприводимого делителя кругового многочлена $Q_{et'}(x)$ равна порядку двойки по модулю et', то есть в рассматриваемом случае меньше mt', значит делитель $h(x) = f(x^{t'})$ будет приводимым, так как его степень равна mt', а поэтому и многочлен $f(x^t) = h(x^{t/t'})$ тоже будет приводимым.

Итак, во всех рассматриваемых случаях (кроме самого первого) многочлен $f(x_t)$ оказывается приводимым.

Из указанного выше вытекает также, что при поиске неприводимых многочленов достаточно ограничиться только многочленами с условием $(n,k) = 1$, которых, как известно, $\phi(n)$ штук. Отметим, что при n имеющем много малых простых делителей, доля таких многочленов может стремиться к нулю, но в среднем, как известно, вероятность того, что трехчлен будет удовлетворять указанному условию асимптотически равна $6/\pi^2$. Если целью является поиск одного неприводимого трехчлена, то, согласно указанному выше, имеет смысл начать с многочленов с большим нечетным (n,k).

Для пятичленов справедливо почти все сказанное выше, за исключением оценки сложности тестирования всех пятичленов, которая становится трудно выводимой.

3.1. Пробные деления

Некоторого ускорения можно добиться с помощью предварительного пробного деления. Например, вначале можно выполнить деление на трехчлен $1 + x + x^2$. Быстрее всего для этого поделить с остатком на $1 + x^3$, для чего достаточно просто заменить все показатели степени в тестируемом многочлене на остатки по модулю 3, и выполнить приведение подобных членов по модулю два, в результате чего и получается искомый остаток по модулю $1 + x^3$. Если он будет равен трехчлену $1 + x + x^2$, то тестируемый многочлен делится на него, и, значит, приводим. Вероятность этого в случае трехчленов равна асимптотически 2/9, а в случае пятичленов – 20/81. Подобный же прием в случае пятичленов и деления на $1 + x + x^2 + x^3 + x^4$ возможен, но не эффективен.

Эффективно, однако пробное деление на $1 + x^7$, остаток в котором находится аналогичным способом (с заменой 3 на 7). После этого полученный остаток проверяется на делимость на $1 + x + x^3$ и $1 + x^2 + x^3$ (это нетривиальные неприводимые делители $1 + x^7$), для чего достаточно его сравнить со всеми кратными им трехчленами не выше 6-й степени, которые суть

$$1 + x^2 + x^6, 1 + x^4 + x^6, 1 + x^2 + x^3, 1 + x + x^3, 1 + x + x^5, 1 + x^4 + x^5,$$

и в случае совпадения с одним из них тестируемый многочлен является приводимым. Заметим, что по существу здесь вычислялся НОД тестируемого многочлена и многочлена $1 + x^7$. Вероятность того, что указанный НОД будет отличен от 1 в случае трехчленов равна 6/49, а полная вероятность результативности одного из двух описанных пробных делений и проверки по теореме Суона равна, согласно формуле включения и исключения и китайской теореме об остатках, $1/2 + 2/9 + 6/49 - 12/441 - 1/9 - 3/49 + 2/147 = 83/126 = 0,658...$ В случае пятичленов указанная вероятность меньше и равна приблизительно 0.347...

В [22] предлагается продолжить эту процедуру и фактически вычислять НОД тестируемого многочлена с двучленами вида $1 +$

$x^{2^m-1}, m = 4, 5, \ldots, 10$. На самом деле, конечно, вначале надо выполнить деление с остатком на $1 + x^{2^m-1}$ аналогично тому, как это делалось выше, а потом полученный остаток (трехчлен или пятичлен степени не выше $2^m - 1$ искать в заранее заготовленном и помещенном в память машины списке трехчленов и пятичленов указанной степени, кратных одному из неприводимых многочленов степени, делящей m (произведение всех таких многочленов, как известно, и равно $1 + x^{2^m-1}$).

Логарифмический поиск в этом списке требует времени $O(m)$ и памяти $O(2^{4m})$ в случае пятичленов и $O(2^{2m})$ в случае трехчленов (что ограничивает m числом 5 в первом и числом 10 во втором случае). Использование алгоритма Евклида или пробного деления на заранее вычисленные неприводимые многочлены степеней, делящих m, ликвидирует проблемы с памятью, но требует в обоих случаях времени в худшем случае $O(2^{2m})$ на один полином.

Для составления указанного списка можно использовать любой из этих подходов, но оценка времени будет умножаться на число проверяемых трехчленов $O(2^{2m})$ или на число проверяемых пятичленов $O(2^{4m})$, что приводит к границе $m \leqslant 8$ в первом и $m \leqslant 6$ во втором случаях. Граница может быть повышена до 10 если тестируются только многочлены специального вида, например $1 + x + x^n$, о чем пойдет речь далее.

Отметим, что указанные приемы сокращают лишь время составления списка неприводимых многочленов с заданными границами степеней, а для тестирования индивидуальных многочленов ускорение достигается лишь с некоторой вероятностью.

Количество неприводимых трехчленов заданной степени невелико, и существуют они не для всех степеней, но очень часто. А вот неприводимые пятичлены заданной степени существуют, видимо, всегда и их количество может быть довольно большим, даже для небольших n. Например, число всех неприводимых пятичленов степени 163 равно примерно 12500 (с точностью до возвратных). Время построения всей таблицы составило около часа работы машины с процессором Pentium III. Небольшой фрагмент этой таблицы приведен

в приложении Г. В приложении Д приводятся примеры неприводимых пятичленов степени 173. Выбор этих двух значений степеней пятичленов обусловлен тем, что для них есть примеры эллиптических кривых (в первом случае несуперсингулярных, а во втором – суперсингулярных), порядки групп которых подсчитаны и содержат в своем разложении на простые множители большие (несколько десятков десятичных цифр) простые числа, что используется в криптографии эллиптических кривых.

4. Построение нормального базиса и вычисление матриц перехода от нормального базиса к стандартному и обратно

Последовательность многочленов $q_k = x^{p^k} \bmod q$, рассматриваемая как последовательность элементов поля $GF(p^n)$, реализованного как кольцо многочленов с операциями по модулю неприводимого многочлена q, образует нормальный базис, причем указанные многочлены задают его разложение по стандартному полиномиальному базису $\{1, x, x^2, \ldots, x^{n-1}\}$, которое, таким образом, получается как бесплатное приложение. Однако, система многочленов $\{q_k\}$ может быть линейно зависимой, и поэтому не образовывать базиса. Для проверки ее на линейную независимость достаточно записать вектора коэффициентов этих многочленов в виде квадратной матрицы и привести ее к диагональному виду стандартной алгебраической процедурой, имеющей сложность по порядку n^3. Причем в этой процедуре можно обрабатывать строки матрицы последовательно, а именно в порядке вычисления коэффициентов многочленов $\{q_k\}$ по формуле $q_{k+1} = q_k^p \bmod q$, и в случае нарушения «диагональности» матрицы сразу прекращать дальнейшие вычисления, если нашей целью является не просто нахождение неприводимого полинома, а непременно полинома, порождающего нормальный базис. В худшем случае указанный алгоритм является кубическим, однако можно вначале целиком построить упомянутую матрицу, а потом диагонализировать ее

и одновременно с этим обратить ее (если она невырождена), найдя при этом разложение стандартного базиса $\{1, x, x^2, \ldots, x^{n-1}\}$ через построенный нормальный. Для этого известен работающий в любом конечном поле алгоритм [11] сложности $O(n^3 / \log n)$.

Известны теоретически еще более быстрые алгоритмы [13] и другие, работающие над любым полем и дающие оценку вида $O(n^{2,4})$, однако они очень сложны и начинают превосходить стандартные алгоритмы только для n порядка тысяч.

При n порядка нескольких сотен в поле $GF(2)$ практичнее алгоритма [11] использовать вариацию стандартного алгоритма, основанную на использовании операций с 32-битовыми числами.

4.1. О нормальных базисах, соответствующих трехчленам и пятичленам

Пусть $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ – корни трехчлена $f(x) = x^n + ax^m + b$ в поле $GF(p^n)$. Очевидным необходимым условием базисности системы $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ является неравенство нулю суммы $\alpha + \alpha^p + \alpha^{p^2} + \ldots \alpha^{p^{n-1}}$, называемой следом элемента α. Так как по теореме Виета эта сумма противоположна по знаку коэффициенту при x^{n-1} многочлена $f(x)$, то этот коэффициент должен быть отличен от нуля, то есть $f(x)$ должен иметь вид $x^n + ax^{n-1} + b$. Однако эти многочлены надо проверять на нормальность с помощью указанного ниже алгоритма. Количество проверок равно $O(p^2)$ в случае трехчленов, и $O(n^2 p^4)$ в случае пятичленов. Если $p = 2$, нужно проверить только один трехчлен, причем легко видеть, что он не может иметь кратных корней.

В случае поля $GF(2)$ поиск неприводимых трехчленов, порождающих нормальные базисы, нужно вести среди трехчленов вида $x^n + x^{n-1} + 1$. Такие трехчлены будут неприводимыми (или примитивными) одновременно с трехчленами вида $x^n + x + 1$. Встречаются они вначале довольно часто, а потом все реже и реже. В [22] имеется таблица таких трехчленов до степени 30000. Их немного – всего около 30. Наша программа нашла еще 2 таких трехчлена: $1 + x^1 + x^{32767}$

и $1+x^1+x^{34353}$. В приложении В приведена таблица трехчленов указанного вида со степенями не превосходящими 1000. В ней также отмечено, какие из них являются примитивными, а также нормальными (то есть порождающими нормальные базисы). В последнем случае указано число единиц в матрице разложения стандартного полиномиального базиса, соответствующего этому многочлену, по соответствующему нормальному базису.

5. Быстрый алгоритм проверки системы нормального вида на базисность

Пусть $\{\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{n-1}}\}$ – последовательность элементов поля $GF(p^n)$. Известен критерий базисности такой последовательности: необходимо и достаточно, чтобы многочлены x^n-1 и

$$f(x) = \alpha x^{n-1} + \alpha^p x^{n-2} + \alpha^{p^2} x^{n-3} + \ldots + \alpha^{p^{n-1}}$$

были взаимно просты над полем $GF(p^n)$ (а значит, согласно свойствам алгоритма Евклида, и над любым расширением этого поля).

Если применить для вычисления НОД этих многочленов алгоритм Евклида в форме Шенхаге-Моенка то оценка сложности его будет

$$O(M(n))A(n,p)\log n,$$

где $A(n,p)$ – сложность вычислений в поле $GF(p^n)$. Отметим, что константа в знаке O довольно велика (порядка 10). Очевидно, что $A(n,p) = O(M_0(\log p)M(n))$, где $M_0(m)$ – сложность умножения m-разрядных двоичных чисел. Как известно, алгоритм Карацубы дает оценку и для $M_0(m)$ и для $M(m)$ в виде $O(m^{\log_2 3})$, а при больших m лучшую оценку $M_0(m)$ дает алгоритм Шенхаге-Штрассена, а именно $O(m\log m\log\log m)$.

Можно $M(n)$ заменить на $3F(n)+2n$, где $F(n)$ – сложность вычисления преобразования Фурье порядка n в минимальном расширении $GF(p^m)$ поля $GF(p^n)$, содержащем все n корней n-степени из единицы. Тогда $m = nk$, и n должно делить p^m-1, а оценку $A(n,p)$ нужно заменить на $A(m,p)$ или на $O(M(k))A(n,p)$, а при малых k – просто на $O(k)A(n,p)$.

Указанную оценку можно улучшить, отбросив множитель $O(\log n)$, если заметить, что взаимная простота рассматриваемых выше многочленов над полем $GF(p^m)$ равносильна тому, что $f(x)$ не имеет общих корней с x^n-1, значит все его значения в корнях n-й степени из единицы отличны от нуля. Но вычисление всех этих значений в поле $GF(p^m)$ есть не что иное, как вычисление преобразования Фурье F_n, поэтому мы получаем оценку $F(n)A(m,p)$. Разлагая n на простые множители $n = p_1^{\beta_1}\ldots p_r^{\beta_r}$ и применяя алгоритмы Гуда-Томаса и Кули-Тьюки (см. [4]), получаем оценки

$$F(n) = n\left(\frac{F(p_1^{\beta_1})}{p_1^{\beta_1}} + \ldots + \frac{F(p_r^{\beta_r})}{p_r^{\beta_r}}\right) \leqslant$$

$$\leqslant n\left(\beta_1\left(\frac{F(p_1)}{p_1}+1\right) + \ldots + \beta_r\left(\frac{F(p_r)}{p_r}+1\right)\right).$$

В случае гладкого числа n (то есть когда все p_i малы) получается оценка, близкая к $O(n\log n)$.

Если же какое-нибудь p_i велико, то для оценки $F(p_i)$ можно применить основанный на применении китайской теоремы об остатках алгоритм Винограда ([4]) вычисления циклической свертки, разложив $x^{p_i}-1$ на круговые множители

$$\prod_{d|p_i} Q_d,$$

которые будут иметь коэффициенты в поле $GF(p)$ и сравнительно мало ненулевых коэффициентов, и поэтому на них проще делить обычным школьным алгоритмом, потом круговые множители разложить над полем $GF(p^n)$ и делить на них.

Можно также методом Райдера ([4]) свести вычисление $F(p_i)$ к вычислению циклической свертки порядка p_i-1, которая в свою очередь сводится к трехкратному применению преобразования Фурье

порядка $p_i - 1$, которое придется вычислять над полем $GF(p^{m_i})$, где m_i/m – порядок числа p^m по модулю $p_i - 1$.

Отметим, что в некоторых случаях n может быть делителем $p^n - 1$, и значит в качестве $m = nk$ можно взять просто n. Например, возьмем $n = p^{\nu_1} - 1$, тогда порядок p по модулю n равен ν_1, значит $m = \frac{n\nu_1}{(n,\nu_1)}$ – наименьшее общее кратное чисел n и ν_1.

Покажем, что при $p > 2$ можно выбрать ν_1 так, чтобы n делилось на ν_1, тогда $(n, \nu_1) = nu_1$ и m будет равно n. Но $n = p^{\nu_1} - 1$, и для того, чтобы оно делилось на ν_1 можно выбрать $\nu_1 = \nu_2 p^{\nu_2} - 1$, и так далее, и для некоторого s выберем $\nu_s = p - 1$, тогда $\nu_{s-1} = p^{\nu_s} - 1$ будет кратно ν_s и мы получаем бесконечную последовательность n_s, рекуррентно определяемую равенствами $n_s = n_{s-1}(p^{n_{s-1}} - 1)$, $n_1 = p - 1$, такую, что $p^{n_s} - 1$ делится на n_s. Действительно, $p^{n_1} - 1 = (n_1 + 1)^{n_1} - 1$ делится на n_1, и далее проверяется, что

$$p^{n_s} - 1 = p^{n_{s-1}(p^{n_{s-1}} - 1)} - 1$$

делится на $p^{n_{s-1}} - 1$, в частном получается $1 + a + a^2 + \ldots + a^{b-1}$, где $a = p^{n_{s-1}}, b = a - 1$, а это число по модулю b равно сумме b единиц то есть равно нулю по модулю b, значит указанное частное делится на n_{s-1}, так как $b = p^{n_{s-1}} - 1$ делится на n_{s-1} согласно предположению индукции, поэтому $p^{n_s} - 1$ делится на $bn_{s-1} = p^{n_{s-1}} n_{s-1} = n_s$.

Если же $p = 2$, то n не может быть делителем $2^n - 1$. Действительно, пусть q – минимальный простой делитель n, тогда q делит одновременно $2^{q-1} - 1$ и $2^n - 1$, а значит и их НОД, равный $2^{(q-1,n)} - 1 = 1$ так как $(q - 1, n) < q$, делит n и согласно выбору p равно 1. Из доказанного следует, что НОД $(n, 2^n - 1)$ делит n/p. Пример, когда $(n, 2^n - 1) = n/q$ очевиден – просто берем $n = q$. Начиная с $n_1 = q$ определим рекуррентно последовательность равенствами $n_s = n_{s-1}(2^{n_{s-1}} - 1)$, тогда рассуждая по индукции замечаем, что $n_s/q = (2^{n_{s-1}} - 1)n_{s-1}/q$ делит $2^{n_s} - 1 = 2^{n_{s-1}(2^{n_{s-1}} - 1)} - 1$ так как при делении $2^{n_{s-1}(2^{n_{s-1}} - 1)} - 1$ на $(2^{n_{s-1}} - 1)$ (как было проверено раньше) получается $1 + a + a^2 + \ldots + a^{b-1}$, где $a = 2^{n_{s-1}}, b = a - 1$, и это число делится на $b = 2^{n_{s-1}} - 1$, а значит согласно предположению индукции и на n_{s-1}/q, поэтому $2^{n_s} - 1 = 2^{n_{s-1}(2^{n_{s-1}} - 1)} - 1$

делится на $(2^{n_{s-1}} - 1)n_{s-1}/q = n_s/q$. Поэтому для всех членов этой последовательности справедливо, что $(2^{n_s} - 1, n_s) = n_s/q$. Значит, взяв $n = 2^{n_s} - 1$, получаем, что $m = \frac{nn_s}{(n,n_s)} = nq$.

6. Тестирование примитивности неприводимого многочлена

Для проверки построенного неприводимого многочлена на примитивность надо проверить, является ли элемент $x \bmod q$ образующим элементом построенного поля $GF(p^n)$. Для этого достаточно проверить для любого простого делителя s числа $p^n - 1$, что $x^{(p^n-1)/s} \bmod q$ не равен 1. Количество таких делителей не превосходит, как отмечалось выше, по порядку $n/\log n$, а в среднем гораздо меньше. Предположим, что они нам известны, и оценим сложность остальной процедуры.

Заметим, что сложность возведения произвольного элемента поля $GF(p^n)$ в степень $m < p^n$ можно при малых p оценить, используя некоторые результаты об аддитивных цепочках ([5]), как

$$O(n \log m) + O(\frac{M(n) \log m}{\log \log m}).$$

Известно ([5]), что для почти всех m второе слагаемое по порядку нельзя уменьшить.

6.1. Одновременное вычисление нескольких степеней в конечном поле

Допустим, что надо вычислить в поле $GF(p^n)$ k разных степеней $f^{n_i}, i = 1, \ldots, k, \max n_i = m$. Используя некоторые результаты об аддитивных цепочках [9], можно получить оценку сложности этого вычисления в худшем случае

$$O(n \log m) + O(k \frac{\log m}{\log \log m} M(n)).$$

Интересно, что оценка сложности всех используемых при этом возведений в p-ую степень при $k = 0$ $(\log\log m)$ остается по порядку такой же, как будто мы возводили в степень лишь одно число!

Можно также показать, что для почти всех k-членных наборов n_i второе слагаемое по порядку нельзя уменьшить, но можно его оценить снизу по порядку как $\log \nu_p(n_1,\ldots,n_k)M(n)$, где $\nu_p(n_1,\ldots,n_k)$ – число единиц в p-ичном разложении k-мерного вектора $(n_1,\ldots,n_k)$, а также как $(l(n_1,\ldots,n_k) - \log m)M(n)$, где $l(n_1,\ldots,n_k)$ – аддитивная сложность системы чисел $n_1,\ldots,n_k$,

Последний член в указанной оценке можно несколько усилить, используя [9], если обозначить произведение всех чисел n_i через N. Тогда упомянутый член можно заменить на $O(N/\log N + k)$.

6.2. Оценка сложности теста на примитивность в конечном поле

Применяя предыдущую оценку к возникающей при тестировании элемента f поля $GF(p^n)$ на примитивность задаче вычисления системы степеней $f^{(p^n-1)/p_i}$, где p_i – система всех различных простых делителей числа $p^n - 1$, получаем в случае отсутствия у него кратных простых множителей следующую оценку сложности вычисления упомянутой системы в худшем случае:

$$O(n^2) + O(\frac{(k-1)M(n)n}{\log n}).$$

Величину k в худшем случае можно оценить как $n/\log n$, в среднем она гораздо меньше, но не меньше, чем $\log n/\log\log n$. Значит, рассматриваемая часть алгоритма имеет менее чем кубическую сложность.

Самая трудная часть тестирования – разложение на простые множители числа $p^n - 1$. Однако ее достаточно сделать для данных n и p только один раз. Обычно указанные разложения берут из таблиц «Каннингемовского проекта» (см. [28] и [29]) или используют программы типа «Математика». Однако эти таблицы для нашего читателя (как возможно и программы) малодоступны, поэтому нелишне

будет в общих чертах описать, как можно построить соответствующий алгоритм самостоятельно, сопроводив его оценками сложности.

Оценка сложности современных алгоритмов факторизации произвольных чисел в худшем случае имеет вид $c^{\sqrt{n\log n}}$.

Факторизация интересующих нас чисел упрощается, если предварительно разложить $p^n - 1$ на множители $\prod_{d|n} f_d(p)$, где f_d – круговые многочлены, и найти у множителей попарные НОД (но НОД у пар множителей вида $f_{p_1}(p) = p^{p_1} - 1$, $f_{p_2}(p) = p^{p_2} - 1$, где p_i – различные простые делители n, можно не искать, так как он обязательно будет равен 1).

Количество указанных сомножителей равно $d(n)$ – числу всех различных делителей n. Известно, что в худшем случае $d(n) = e^{\log n/\log\log n} = n^{c/\log\log n}$, но в среднем $d(n) = \log n$ согласно классическим теоремам Рамануджана и Дирихле [8].

Вычисление коэффициентов кругового многочлена f_n с помощью известного алгебраического алгоритма, основанного на тождествах

$$f_{np}(x) = f_n(x^p)/f_n(x), (n,p) = 1,$$

$$f_n(x) = f_{p_1\ldots p_r}(x^{p_1^{\alpha_1-1}\ldots p_r^{\alpha_r-1}}), n = p_1^{\alpha_1}\ldots p_r^{\alpha_r},$$

выполняется со сложностью $O(n\log^2 n)$, а всего разложения на круговые многочлены – со сложностью $O(n\log^2 n\log\log n)$.

Задача факторизации несколько упрощается в случае $p = 2$, наиболее важном практически. Распознавание простоты чисел Мерсенна $2^n - 1$ выполняется тестом Люка (см. [5], [12]) со сложностью $O(n^2\log^2 n)$, т. е. меньше кубической. В [12] имеются доступные таблицы факторизации чисел Мерсенна и описания соотвествующих алгоритмов.

Даже если найдены не все простые делители у $2^n - 1$, то все равно приведенный тест позволяет быстро отбрасывать элементы, не являющиеся примитивными корнями. Если же n – само простое и $2^n - 1$ – простое, то любой элемент является примитивным и проверка на примитивность очевидно не нужна.

Если нам уже известен один примитивный элемент в рассматриваемом поле, то можно было бы прологарифмировать тестируемый

элемент по указанному основанию и вычислить НОД найденного логарифма и числа 2^n-1, что делается, как отмечалось выше, быстрее, чем с кубической сложностью. Однако логарифмирование выполняется с большой сложностью, так, алгоритм [27] имеет сложность $c^{n^{1/3}\log^{2/3} n}$, где c – некоторая константа.

В случае поиска примитивных многочленов f со старшим коэффициентом единица над произвольным конечным полем $GF(q)$ можно, как отмечено в [7], сначала проверить, что $(-1)^n f(0)$ примитивный элемент в поле коэффициентов, потом проверить, что $x^r = (-1)^n f(0) \bmod f(x)$, где $r = (q^n-1)/(q-1)$ (для чего достаточно не более чем $n-1$ умножений по модулю $f(x)$ и n возведений в q-ю степень по тому же модулю, которые выполняются проще умножения), и для любого простого делителя s числа r многочлен $x^{r/s} \bmod f(x)$ имеет положительную степень. В случае, если операции в поле $GF(q)$ затабулированы и хранятся в оперативной памяти машины, такой подход несколько ускоряет вычисления. Его можно применять и для поиска примитивного элемента в поле $GF(p^n)$, если выбрать $q = p^m$, где m – небольшой делитель n.

Приложение

А. Таблица всех неприводимых трехчленов степени n, $151 \leqslant n \leqslant 175$

$1+x^3+x^{151}$	$1+x^{67}+x^{151}$	$1+x^{120}+x^{151}$	$1+x^{62}+x^{155}$
$1+x^9+x^{151}$	$1+x^{70}+x^{151}$	$1+x^{136}+x^{151}$	$1+x^{93}+x^{155}$
$1+x^{15}+x^{151}$	$1+x^{81}+x^{151}$	$1+x^{142}+x^{151}$	$1+x^{9}+x^{156}$
$1+x^{31}+x^{151}$	$1+x^{84}+x^{151}$	$1+x^{148}+x^{151}$	$1+x^{11}+x^{156}$
$1+x^{39}+x^{151}$	$1+x^{85}+x^{151}$	$1+x^{1}+x^{153}$	$1+x^{21}+x^{156}$
$1+x^{43}+x^{151}$	$1+x^{88}+x^{151}$	$1+x^{8}+x^{153}$	$1+x^{39}+x^{156}$
$1+x^{46}+x^{151}$	$1+x^{100}+x^{151}$	$1+x^{145}+x^{153}$	$1+x^{57}+x^{156}$
$1+x^{51}+x^{151}$	$1+x^{105}+x^{151}$	$1+x^{152}+x^{153}$	$1+x^{61}+x^{156}$
$1+x^{63}+x^{151}$	$1+x^{108}+x^{151}$	$1+x^{15}+x^{154}$	$1+x^{63}+x^{156}$
$1+x^{66}+x^{151}$	$1+x^{112}+x^{151}$	$1+x^{139}+x^{154}$	$1+x^{65}+x^{156}$

$1+x^{91}+x^{156}$	$1+x^{155}+x^{160}$	$1+x^{35}+x^{167}$	$1+x^{11}+x^{170}$
$1+x^{93}+x^{156}$	$1+x^{18}+x^{161}$	$1+x^{59}+x^{167}$	$1+x^{23}+x^{170}$
$1+x^{95}+x^{156}$	$1+x^{39}+x^{161}$	$1+x^{77}+x^{167}$	$1+x^{147}+x^{170}$
$1+x^{99}+x^{156}$	$1+x^{60}+x^{161}$	$1+x^{90}+x^{167}$	$1+x^{159}+x^{170}$
$1+x^{117}+x^{156}$	$1+x^{101}+x^{161}$	$1+x^{108}+x^{167}$	$1+x^{1}+x^{172}$
$1+x^{135}+x^{156}$	$1+x^{122}+x^{161}$	$1+x^{132}+x^{167}$	$1+x^{7}+x^{172}$
$1+x^{145}+x^{156}$	$1+x^{143}+x^{161}$	$1+x^{161}+x^{167}$	$1+x^{81}+x^{172}$
$1+x^{147}+x^{156}$	$1+x^{27}+x^{162}$	$1+x^{34}+x^{169}$	$1+x^{91}+x^{172}$
$1+x^{31}+x^{159}$	$1+x^{63}+x^{162}$	$1+x^{42}+x^{169}$	$1+x^{165}+x^{172}$
$1+x^{34}+x^{159}$	$1+x^{81}+x^{162}$	$1+x^{57}+x^{169}$	$1+x^{171}+x^{172}$
$1+x^{40}+x^{159}$	$1+x^{99}+x^{162}$	$1+x^{84}+x^{169}$	$1+x^{13}+x^{174}$
$1+x^{119}+x^{159}$	$1+x^{135}+x^{162}$	$1+x^{85}+x^{169}$	$1+x^{57}+x^{174}$
$1+x^{125}+x^{159}$	$1+x^{37}+x^{166}$	$1+x^{112}+x^{169}$	$1+x^{117}+x^{174}$
$1+x^{128}+x^{159}$	$1+x^{129}+x^{166}$	$1+x^{127}+x^{169}$	$1+x^{161}+x^{174}$
$1+x^{5}+x^{160}$	$1+x^{6}+x^{167}$	$1+x^{135}+x^{169}$	$1+x^{6}+x^{175}$

Б. Таблица всех (с точностью до возвратных) неприводимых трехчленов степени n, $2000 < n \leqslant 2100$

$1+x^{169}+x^{2001}$	$1+x^{849}+x^{2009}$	$1+x^{549}+x^{2023}$	$1+x^{909}+x^{2033}$
$1+x^{475}+x^{2001}$	$1+x^{459}+x^{2010}$	$1+x^{751}+x^{2023}$	$1+x^{143}+x^{2034}$
$1+x^{511}+x^{2001}$	$1+x^{819}+x^{2010}$	$1+x^{274}+x^{2025}$	$1+x^{71}+x^{2036}$
$1+x^{752}+x^{2001}$	$1+x^{42}+x^{2015}$	$1+x^{289}+x^{2025}$	$1+x^{195}+x^{2036}$
$1+x^{860}+x^{2001}$	$1+x^{344}+x^{2015}$	$1+x^{859}+x^{2025}$	$1+x^{783}+x^{2036}$
$1+x^{441}+x^{2004}$	$1+x^{558}+x^{2015}$	$1+x^{1001}+x^{2025}$	$1+x^{857}+x^{2036}$
$1+x^{533}+x^{2004}$	$1+x^{714}+x^{2015}$	$1+x^{1003}+x^{2025}$	$1+x^{155}+x^{2039}$
$1+x^{917}+x^{2006}$	$1+x^{992}+x^{2015}$	$1+x^{475}+x^{2026}$	$1+x^{651}+x^{2039}$
$1+x^{205}+x^{2007}$	$1+x^{330}+x^{2017}$	$1+x^{93}+x^{2028}$	$1+x^{840}+x^{2039}$
$1+x^{314}+x^{2007}$	$1+x^{540}+x^{2017}$	$1+x^{301}+x^{2028}$	$1+x^{947}+x^{2039}$
$1+x^{427}+x^{2007}$	$1+x^{589}+x^{2017}$	$1+x^{723}+x^{2028}$	$1+x^{735}+x^{2041}$
$1+x^{523}+x^{2007}$	$1+x^{81}+x^{2020}$	$1+x^{793}+x^{2028}$	$1+x^{771}+x^{2041}$
$1+x^{737}+x^{2007}$	$1+x^{325}+x^{2020}$	$1+x^{831}+x^{2028}$	$1+x^{45}+x^{2044}$
$1+x^{859}+x^{2007}$	$1+x^{543}+x^{2020}$	$1+x^{845}+x^{2028}$	$1+x^{85}+x^{2044}$
$1+x^{889}+x^{2007}$	$1+x^{945}+x^{2020}$	$1+x^{847}+x^{2028}$	$1+x^{289}+x^{2044}$
$1+x^{54}+x^{2009}$	$1+x^{349}+x^{2022}$	$1+x^{881}+x^{2028}$	$1+x^{639}+x^{2044}$
$1+x^{150}+x^{2009}$	$1+x^{409}+x^{2022}$	$1+x^{386}+x^{2031}$	$1+x^{655}+x^{2044}$
$1+x^{710}+x^{2009}$	$1+x^{165}+x^{2023}$	$1+x^{400}+x^{2031}$	$1+x^{789}+x^{2044}$
$1+x^{771}+x^{2009}$	$1+x^{430}+x^{2023}$	$1+x^{881}+x^{2033}$	$1+x^{3}+x^{2047}$

$1+x^{66}+x^{2047}$
$1+x^{165}+x^{2047}$
$1+x^{411}+x^{2047}$
$1+x^{495}+x^{2047}$
$1+x^{511}+x^{2047}$
$1+x^{817}+x^{2047}$
$1+x^{124}+x^{2049}$
$1+x^{140}+x^{2049}$
$1+x^{433}+x^{2049}$
$1+x^{523}+x^{2049}$
$1+x^{934}+x^{2049}$
$1+x^{323}+x^{2052}$
$1+x^{589}+x^{2052}$
$1+x^{627}+x^{2052}$
$1+x^{201}+x^{2054}$
$1+x^{497}+x^{2054}$
$1+x^{11}+x^{2055}$
$1+x^{392}+x^{2055}$
$1+x^{245}+x^{2057}$
$1+x^{633}+x^{2057}$
$1+x^{343}+x^{2058}$
$1+x^{427}+x^{2058}$
$1+x^{591}+x^{2058}$
$1+x^{667}+x^{2058}$
$1+x^{387}+x^{2060}$
$1+x^{735}+x^{2060}$
$1+x^{981}+x^{2060}$
$1+x^{987}+x^{2060}$
$1+x^{48}+x^{2063}$
$1+x^{353}+x^{2063}$
$1+x^{570}+x^{2063}$
$1+x^{674}+x^{2063}$
$1+x^{719}+x^{2063}$
$1+x^{770}+x^{2063}$
$1+x^{97}+x^{2065}$
$1+x^{382}+x^{2065}$
$1+x^{918}+x^{2065}$
$1+x^{71}+x^{2066}$
$1+x^{237}+x^{2070}$
$1+x^{253}+x^{2073}$
$1+x^{557}+x^{2073}$
$1+x^{231}+x^{2074}$
$1+x^{583}+x^{2074}$
$1+x^{851}+x^{2076}$
$1+x^{897}+x^{2076}$
$1+x^{933}+x^{2076}$
$1+x^{35}+x^{2079}$
$1+x^{134}+x^{2079}$
$1+x^{283}+x^{2079}$
$1+x^{581}+x^{2079}$
$1+x^{590}+x^{2079}$
$1+x^{721}+x^{2079}$
$1+x^{755}+x^{2079}$
$1+x^{796}+x^{2079}$
$1+x^{854}+x^{2079}$
$1+x^{928}+x^{2079}$
$1+x^{959}+x^{2079}$
$1+x^{998}+x^{2079}$
$1+x^{467}+x^{2081}$
$1+x^{662}+x^{2081}$
$1+x^{980}+x^{2081}$
$1+x^{523}+x^{2082}$
$1+x^{261}+x^{2086}$
$1+x^{673}+x^{2086}$
$1+x^{141}+x^{2087}$
$1+x^{225}+x^{2087}$
$1+x^{569}+x^{2087}$
$1+x^{737}+x^{2087}$
$1+x^{150}+x^{2089}$
$1+x^{349}+x^{2089}$
$1+x^{357}+x^{2089}$
$1+x^{846}+x^{2089}$
$1+x^{853}+x^{2089}$
$1+x^{909}+x^{2089}$
$1+x^{645}+x^{2094}$
$1+x^{933}+x^{2094}$
$1+x^{256}+x^{2095}$
$1+x^{457}+x^{2095}$
$1+x^{607}+x^{2095}$
$1+x^{691}+x^{2095}$
$1+x^{119}+x^{2097}$
$1+x^{19}+x^{2098}$
$1+x^{35}+x^{2100}$
$1+x^{49}+x^{2100}$
$1+x^{61}+x^{2100}$
$1+x^{193}+x^{2100}$
$1+x^{225}+x^{2100}$
$1+x^{325}+x^{2100}$
$1+x^{385}+x^{2100}$
$1+x^{435}+x^{2100}$
$1+x^{511}+x^{2100}$
$1+x^{583}+x^{2100}$
$1+x^{635}+x^{2100}$
$1+x^{637}+x^{2100}$
$1+x^{675}+x^{2100}$
$1+x^{923}+x^{2100}$
$1+x^{975}+x^{2100}$
$1+x^{1009}+x^{2100}$

В. Таблица всех неприводимых трехчленов степени $n \leqslant 1000$, порождающих нормальные базисы

Трехчлен	Число единиц в матрице нормального базиса	Свойство примитивности
$1+x^1+x^2$	3	
$1+x^2+x^3$	5	
$1+x^3+x^4$	9	
$1+x^5+x^6$	15	
$1+x^6+x^7$	23	
$1+x^8+x^9$	33	
$1+x^{14}+x^{15}$	107	примитивный
$1+x^{21}+x^{22}$	185	примитивный

Трехчлен	Число единиц в матрице нормального базиса	Свойство примитивности
$1+x^{27}+x^{28}$	331	
$1+x^{29}+x^{30}$	361	
$1+x^{45}+x^{46}$	915	
$1+x^{59}+x^{60}$	1709	примитивный
$1+x^{62}+x^{63}$	1915	примитивный
$1+x^{126}+x^{127}$		примитивный
$1+x^{152}+x^{153}$		примитивный
$1+x^{171}+x^{172}$	13995	
$1+x^{302}+x^{303}$	44823	
$1+x^{470}+x^{471}$	109433	
$1+x^{531}+x^{532}$	139431	
$1+x^{864}+x^{865}$	370059	
$1+x^{899}+x^{900}$	400731	

Г. Фрагмент таблицы неприводимых пятичленов степени $n = 163$

$1+x^3+x^6+x^7+x^{163}$
$1+x^4+x^{58}+x^{79}+x^{163}$
$1+x^2+x^{27}+x^{83}+x^{163}$
$1+x^{42}+x^{51}+x^{85}+x^{163}$
$1+x^2+x^{82}+x^{86}+x^{163}$
$1+x^{15}+x^{54}+x^{88}+x^{163}$
$1+x^{10}+x^{41}+x^{89}+x^{163}$
$1+x^7+x^{68}+x^{90}+x^{163}$
$1+x^8+x^{22}+x^{92}+x^{163}$
$1+x^{86}+x^{91}+x^{92}+x^{163}$
$1+x^2+x^{11}+x^{93}+x^{163}$
$1+x^{25}+x^{95}+x^{96}+x^{163}$
$1+x^2+x^8+x^{97}+x^{163}$
$1+x^{17}+x^{97}+x^{98}+x^{163}$
$1+x^6+x^{10}+x^{99}+x^{163}$
$1+x^6+x^{18}+x^{99}+x^{163}$
$1+x^8+x^{26}+x^{99}+x^{163}$
$1+x^6+x^{58}+x^{99}+x^{163}$
$1+x^{14}+x^{77}+x^{100}+x^{163}$
$1+x^{21}+x^{77}+x^{100}+x^{163}$
$1+x^{23}+x^{77}+x^{102}+x^{163}$
$1+x^{24}+x^{49}+x^{104}+x^{163}$
$1+x^{101}+x^{114}+x^{124}+x^{163}$
$1+x^{94}+x^{135}+x^{136}+x^{163}$
$1+x^2+x^9+x^{137}+x^{163}$
$1+x^{115}+x^{137}+x^{143}+x^{163}$
$1+x^{117}+x^{137}+x^{143}+x^{163}$
$1+x^{81}+x^{85}+x^{144}+x^{163}$
$1+x^{25}+x^{86}+x^{144}+x^{163}$
$1+x^{10}+x^{87}+x^{144}+x^{163}$
$1+x^2+x^{38}+x^{145}+x^{163}$
$1+x^{26}+x^{54}+x^{145}+x^{163}$
$1+x^9+x^{65}+x^{145}+x^{163}$
$1+x^{17}+x^{65}+x^{145}+x^{163}$
$1+x^5+x^{58}+x^{153}+x^{163}$
$1+x^{33}+x^{58}+x^{153}+x^{163}$

$1+x^{119}+x^{154}+x^{156}+x^{163}$	$1+x^{81}+x^{157}+x^{162}+x^{163}$
$1+x^{104}+x^{114}+x^{157}+x^{163}$	$1+x^{17}+x^{158}+x^{162}+x^{163}$
$1+x^{135}+x^{146}+x^{157}+x^{163}$	$1+x^{23}+x^{158}+x^{162}+x^{163}$
$1+x^{4}+x^{63}+x^{158}+x^{163}$	$1+x^{12}+x^{159}+x^{162}+x^{163}$
$1+x^{21}+x^{157}+x^{162}+x^{163}$	$1+x^{77}+x^{161}+x^{162}+x^{163}$
$1+x^{56}+x^{157}+x^{162}+x^{163}$	$1+x^{155}+x^{161}+x^{162}+x^{163}$

Д. Фрагмент таблицы неприводимых пятичленов степени $n = 173$

$1+x^{2}+x^{5}+x^{8}+x^{173}$	$1+x^{74}+x^{75}+x^{79}+x^{173}$
$1+x^{5}+x^{7}+x^{10}+x^{173}$	$1+x^{76}+x^{79}+x^{80}+x^{173}$
$1+x^{10}+x^{11}+x^{16}+x^{173}$	$1+x^{76}+x^{83}+x^{84}+x^{173}$
$1+x^{15}+x^{16}+x^{17}+x^{173}$	$1+x^{76}+x^{85}+x^{86}+x^{173}$
$1+x^{20}+x^{21}+x^{23}+x^{173}$	$1+x^{83}+x^{87}+x^{89}+x^{173}$
$1+x^{20}+x^{23}+x^{29}+x^{173}$	$1+x^{83}+x^{87}+x^{94}+x^{173}$
$1+x^{25}+x^{26}+x^{30}+x^{173}$	$1+x^{87}+x^{88}+x^{97}+x^{173}$
$1+x^{30}+x^{31}+x^{34}+x^{173}$	$1+x^{87}+x^{89}+x^{98}+x^{173}$
$1+x^{30}+x^{33}+x^{35}+x^{173}$	$1+x^{88}+x^{91}+x^{99}+x^{173}$
$1+x^{32}+x^{36}+x^{38}+x^{173}$	$1+x^{96}+x^{98}+x^{101}+x^{173}$
$1+x^{32}+x^{38}+x^{41}+x^{173}$	$1+x^{98}+x^{99}+x^{106}+x^{173}$
$1+x^{36}+x^{38}+x^{44}+x^{173}$	$1+x^{100}+x^{102}+x^{109}+x^{173}$
$1+x^{36}+x^{42}+x^{45}+x^{173}$	$1+x^{100}+x^{103}+x^{110}+x^{173}$
$1+x^{41}+x^{42}+x^{46}+x^{173}$	$1+x^{106}+x^{109}+x^{111}+x^{173}$
$1+x^{43}+x^{47}+x^{48}+x^{173}$	$1+x^{107}+x^{113}+x^{116}+x^{173}$
$1+x^{48}+x^{50}+x^{51}+x^{173}$	$1+x^{110}+x^{114}+x^{117}+x^{173}$
$1+x^{51}+x^{52}+x^{55}+x^{173}$	$1+x^{110}+x^{117}+x^{122}+x^{173}$
$1+x^{51}+x^{56}+x^{63}+x^{173}$	$1+x^{116}+x^{122}+x^{123}+x^{173}$
$1+x^{55}+x^{59}+x^{66}+x^{173}$	$1+x^{122}+x^{123}+x^{125}+x^{173}$
$1+x^{55}+x^{59}+x^{67}+x^{173}$	$1+x^{125}+x^{126}+x^{130}+x^{173}$
$1+x^{63}+x^{68}+x^{69}+x^{173}$	$1+x^{127}+x^{131}+x^{132}+x^{173}$
$1+x^{63}+x^{70}+x^{73}+x^{173}$	$1+x^{128}+x^{131}+x^{137}+x^{173}$
$1+x^{67}+x^{74}+x^{75}+x^{173}$	$1+x^{132}+x^{135}+x^{141}+x^{173}$
$1+x^{72}+x^{75}+x^{77}+x^{173}$	$1+x^{134}+x^{139}+x^{142}+x^{173}$

Список литературы

[1] Ахо А., Хопкрофт Д., Ульман Д. Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

[2] Берлекемп Е. Алгебраическая теория кодирования. М.: Мир, 1971.

[3] Биркгоф Г., Барти Т. Современная прикладная алгебра. М.: Мир, 1972.

[4] Блейхут Р. Быстрые алгоритмы цифровой обработки сигналов. М.: Мир, 1989.

[5] Кнут Д. Искусство программирования на ЭВМ. Т. 2. М.: Мир, 1976.

[6] Жельников В. Криптография от папируса до компьютера. М.: ABF, 1996.

[7] Лидл Р., Нидеррейтер Х. Конечные поля. М.: Мир, 1988.

[8] Прахар К. Распределение простых чисел. М.: Мир, 1967.

[9] Gashkov S., Kochergin V. On addition chains of vectors, gate circuits, and the complexity of computation of power // Syberian Advances in Mathematics. 1994. V. 4. №4. 1-16.

[10] Карацуба А.А., Офман Ю.П. Умножение многозначных чисел на автоматах // ДАН СССР. 1962. 145. №2. 293-294.

[11] Коновальцев И.В. Об одном алгоритме решения систем линейных уравнений в конечных полях // Проблемы кибернетики. М.: Физматгиз, 1967. Вып. 19. 269-274.

[12] Уильямс Х. Проверка чисел на простоту с помощью вычислительных машин // Кибернетический сборник. М.: Мир, 1986. Вып. 23.

[13] Штрассен В. Алгоритм Гаусса не оптимален // Кибернетический сборник. М.: Мир, 1971. Вып. 7.

[14] Schonhage A. Schnelle berechnung von kettenbruchentwicklungen // Acta Informatica 1. 1971. P. 139-144.

[15] Schonhage A. Schnelle Multiplikation von Polynomen ueber Koerpern der Charakteristik 2 // Acta Informatica 7. 1977. P. 395-398.

[16] Moenk R. Fast algorithm of GCD's // Proceedings of the 5th Annual ACM Symposium on Theory of Computing. 1973. P. 142-151.

[17] Swift J.D. Construction of Galois Fields of Charasteristic Two and Irreducible Polynomials // Math. Comp. 1960. Vol. 14, 70. 99-103.

[18] Watson E.J. Primitive Polynomials (Mod 2) // Math. Comp. 1962. Vol. 16, 79. 368-369.

[19] Alanen J.D., Knuth D.E. Tables of finite fields // Sankhya. 1964. Ser. A 26. 305-328.

[20] Stahmke W. Primitive Binary Polynomials // Math. Comp. 1973. Vol. 27, 124. 977-980.

[21] Zirler N., Brillhart J. On primitive trinomials (mod 2) // Inform. Contr. 1968. 13. 541-554.

[22] Zirler N. On $x^n + x + 1$ over $GF(2)$ // Inform. Contr. 1970. 16. 502-505.

[23] Zirler N., Brillhart J. On primitive trinomials // Inform. Contr. 1969. 14. 566-569.

[24] Zivkovic M. A table of primitive binary polynomials // Math. Comp. 1994. 62. 385-386.

[25] Zivkovic M. A table of primitive binary polynomials. II // Math. Comp. 1994. 63. 301-306.

[26] Hansen T., Mullen G. Primitive polynomials over finite fields // Math. Comp. 1992. 59. 639-643.

[27] Coppersmith D. Fast evaluation of logarithms in fields of characteristic two // IEEE Trans. Inform. Theory. 1984. 30. 587-594.

[28] Brillhart J., Lehmer D., Selfridge J., Tuckerman B., Wagstaff S. Jr. Factorization of $b^n \pm 1, b = 2, 3, 5, 6, 7, 10, 11, 12$ up to high powers 2nd ed. // Contemp. Math. Providence, RI: Amer. Math. Soc., 1988. Vol. 22.

[29] Wagstaff S. Jr. Update 2.6 to the second edition of factorization of $b^n \pm 1$. 1993.

[30] Menezes A., van Oorshot P., Vanstone S. Handbook of applied cryptography. CRC Press, 1999.

Об автоматной модели защищенных компьютерных систем

А.В. Галатенко

В работе строится вероятностная модель компьютерной системы, вводится понятие безопасности для нее и приводятся условия, при соблюдении которых система является безопасной.

Введение

В работе [1] компьютерная система рассматривается как конечный автомат, что позволило распространить некоторые факты из теории автоматов на эти системы. Отмечается, что дальнейшее обобщение таких систем осуществляется за счет рассмотрения вероятностных автоматов. Ниже приводятся результаты такого обобщения.

1. Основные понятия и результаты

Пусть Σ – конечное множество, Σ^* – множество всех конечных слов над Σ, и ε – пустое слово.

Под вероятностным автоматом понимается тройка $\mathrm{A} = (S, \Sigma, \delta)$, где S и Σ суть конечные множества (состояния и входной алфавит, соответственно), а δ – функция, определенная на множестве $S \times \Sigma$ и принимающая в качестве значений вероятностные меры на множестве S (обозначим множество таких мер через μ).