Д К	МК			УТ		
	к=4	к=8	к=16	к=4	к=8	к=16
0000	0	0	0	0	0	0
0001	1	1	1	5*	5*	5*
0010	2	2	2	10*	10*	10*
0011	3	3	3	15	15	15
0100	–	4	4	–	20*	20*
0101	–	5	5	–	25	25
0110	–	6	6	–	30	30
0111	–	7	7	–	35	35
1000	–	–	8	–	–	40*
1001	–	–	9	–	–	45
1010	–	–	10	–	–	50
1011	–	–	11	–	–	55
1100	–	–	12	–	–	60
1101	–	–	13	–	–	65
1110	–	–	14	–	–	70
1111	–	–	15	–	–	75

быстродействия и от значности, которая используется в УВО. В целях сравнения и анализа этих отличий возможно совмещение таблиц истинности и математических моделей комбинационных схем для k=4, 8, 16. Выбор значений k связан, во-первых, с тем, что УВО преобразуют позиционные двузначные коды в k-значные, и наоборот – k-значные в двузначные. Такое преобразование, как уже отмечалось ранее, удобнее всего осуществлять, если $k=2^n$, где n – число разрядов двузначного кода, тогда коэффициент сжатия $e=\log_2 k$ и для k=4 - e=2, для k=8 - e=3, для k=16 - e=4. Во–вторых, выбор верхнего значения k связан с исследованиями [3] взаимосвязи значности и жесткости допуска на шаг квантования в k-значных микроэлектроник-структурах, которые ограничили верхнее значение на указанном уровне. Дальнейшее повышение значности означает перемещение из области k-значных структур в область аналого-цифровых устройств, где методы построения и принципы действия несколько иные.

Входные дешифраторы предельного быстродействия синтезированы по структуре одноступенчатых линейных дешифраторов, а выходные – по структуре усеченного параллельно-последовательного сумматора, поскольку его входные сигналы никогда не пробегают всего множества значений. Последовательность в алгоритме и структуре возникает в силу необходимости анализировать во время суммирования многоразрядных чисел межразрядные связи и переносы.

Регулярность значений таблиц истинности и структур дешифраторов при k=4,8,16 позволяет построить два максимальных варианта входного и выходного дешифраторов для k=16, которые состоят из однотипных субблоков дешифрации. Дополнительно обратим внимание на то, что полученная однородность структур есть проявление и следствие применения принципа симбиоза: синтезированные структуры как входного, так и выходного дешифраторов в максимальном варианте k=16 включают в состав все субблоки дешифрации предыдущих значимостей.

Литература: 1. *Бондаренко М.Ф., Коноплянко З.Д., Четвериков Г.Г.* Основи теорії синтезу надшвидкодіючих структур мовних систем штучного інтелекту. К.: ІЗМН, 1997.264 с. 2. *Коноплянко З.Д.* Принципы построения многозначных систем искусственного интелекта // Проблемы бионики. Х.: Основа, 1990. Вып. 45. С. 27–35. 3. *Четвериков Г.Г.* Многозначные структуры (анализ, сравнение, синтез, обобщение). Ч.1: Учеб. пособие. К.: ИСМО, 1997.192 с. 4. *А.с. 1241483 СССР, МКИ НОЗМ 7/00.* Модуль для преобразования кодов, заданных логическими уравнениями / Г.Г. Четвериков, М.Ф. Бондаренко, Ю.П. Шабанов-Кушнаренко (СССР) // Открытия. Изобретения. 1984. №3844735/24–24.

Рецензент: д-р физ.-мат. наук, проф. Яковлев С.В.

Четвериков Григорий Григорьевич, канд. техн. наук, доцент кафедры ПО ЭВМ ХТУРЭ. Научные интересы: разработка теории и практика использования методов синтеза многозначных структур языковых систем искусственного интеллекта. Адрес: 310726, Украина, Харьков, пр. Ленина, 14, тел. 40-94-46, 27-97-48.

Стороженко Александра Владимировна, асистент кафедры экономики и менеджмента ХТУРЭ. Научные интересы: мнгозначные структуры в системах искусственного интеллекта. Адрес: 310726, Украина, Харьков, пр. Ленина, 14, тел. 40-94-46, 43-49-02.

Ревенчук Илона Анатольевна, аспирант кафедры ПО ЭВМ ХТУРЭ. Научные интересы: архитектура и принципы построения цифровых многозначных элементов и структур. Адрес: 310726, Украина, Харьков, пр. Ленина, 14, тел. 40-94-46.

Бавыкин Виктор Николаевич: асприрант кафедры ПО ЭВМ ХТУРЭ. Научные интересы: многозначные структуры в системах искусственного интеллекта. Адрес: 310726, Украина, Харьков, пр. Ленина, 14, тел. 40-94-46.

УДК 681.3.06: 519.248.681

ПОСТРОЕНИЕ СИЛЬНЫХ ПРОСТЫХ ЧИСЕЛ ДЛЯ НЕСИММЕТРИЧНЫХ КРИПТОСИСТЕМ

ГОРБЕНКО И.Д., ЖИЛИН О.В.

Работа посвящена проблеме построения аналитически простых чисел специального вида. Требуемый вид результирующего числа N обеспечивается заранее известным разложением N±1. Проведены необходимые теоретические исследования, и на их основании разработан и реализован алгоритм построения простых чисел нужного вида. Выполнены экспериментальные исследования и анализ вычислительной сложности разработанных алгоритмов.

Введение

В несимметричных криптографических системах в качестве общесистемного параметра используются большие простые числа. В наиболее широко распространенном алгоритме RSA [1] в качестве общего модуля используется число N – произведение двух “сильных” простых чисел P и Q. Для реализации алгоритмов шифрования и цифровой подписи класса Эль-Гамаля [2], а также протокола распространения ключей Диффи-Хелмана [3] необходимы простые числа специального вида.

Методы получения простых чисел можно разделить на три группы. К первой относятся методы, основанные на вероятностных тестах. Наиболее широко применяются тесты Соловея-Штрассена [4], Рабина-Миллера [5], Бейли-Вагстаффа [6]. После применения каждого теста можно либо с определенной вероятностью утверждать, что число про-

стое, либо убедиться, что оно составное. Чтобы получить простое число с требуемой вероятностью, применяют ряд циклов одного или нескольких тестов. Вычислительная сложность вероятностных тестов приемлема для практических приложений, однако полученные числа являются псевдопростыми, т.е. существует вероятность P_s, что каждое из них может быть составным. В этом состоит основное преимущество и основной недостаток применения вероятностных методов.

Ко второй группе относятся методы гипотез. Примером является метод Миллера, предложенный в [6], однако не получивший широкого распространения по двум причинам. Первая из них – его большая вычислительная сложность. Кроме того, одно из предположений метода базируется на не доказанной до сих пор обобщенной гипотезе Римана, что снижает доверие к нему. Тем не менее, были разработаны некоторые усовершенствования метода Миллера.

Рассматриваемый в данной статье метод относится к третьей группе – к аналитическим методам. В случае их применения доказательством простоты получаемых чисел является сам алгоритм их построения. Многие аналитические методы не получили широкого распространения из-за большой вычислительной сложности. Тем не менее, в случае, когда простое число используется в качестве общесистемного параметра, изменяемого достаточно редко, с точки зрения стойкости системы от криптологических атак на общесистемные параметры часто более предпочтительно затратить некоторое время на генерацию аналитически простого числа. Приводимый ниже метод предоставляет возможность получать не только достоверно простые числа, но и за приемлемое время, т.е. обладает ограниченной вычислительной сложностью. К тому же имеется возможность построения простых чисел специального вида, что часто является необходимым условием для реализации криптографических систем. Данный метод был предложен Янником Саутером в [7]. Для проверки чисел на простоту он предложил использовать заранее известное разложение P±1.

1. Теоретические основы метода

Задачей метода является генерация такой пары простых чисел P и Q, чтобы сложность факторизации модуля N=P·Q была максимальной независимо от применяемых известных алгоритмов факторизации. Этого можно достигнуть, если P и Q будут сильными простыми числами [8].

Число P называется сильным простым числом, если:

$$\begin{aligned} p-1 &= r_j \cdot t_1, \\ p+1 &= s_\nu \cdot t_2, \\ r_j-1 &= h_\xi \cdot t_3, \end{aligned} \qquad (1)$$

где r_j, s_ν, h_ξ – простые числа соответствующей размерности; t_1, t_2, t_3 – четные целые числа.

Известные алгоритмы факторизации, такие как ρ-алгоритм Полларда [9] и метод эллиптических кривых Ленстры [10], факторизацию N выполняют путем подбора делителей P-1 и Q-1. В случае же, когда P и Q – сильные простые числа, факторизация N затруднительна тем больше, чем большие простые числа являются делителями P±1 и Q±1. В других случаях факторизация производится без предположений о делителях. При этом гарантией стойкости числа N является его длина.

Для того чтобы получаемое простое число удовлетворяло требованиям (1), предлагается генерировать числа с заранее известным частичным разложением на множители чисел N±1. Задавая достаточно большие простые делители и применяя предлагаемый метод несколько раз можно обеспечить выполнение требований (1). Кроме того, существует аналитический тест, использующий частичное разложение N±1. Приведем основные определения, используемые в математическом описании метода.

Определение 1. Символ Кронекера. Пусть i и j – целые числа. Символ Кронекера, обозначенный δ_i^j, равен 1, если j=i, иначе — 0.

Определение 2. Символ Лежандра. Пусть a – целое, p – простое. Символ Лежандра $L(a,p)$ равен 1, если существует b, такое что $b^2 \equiv a \pmod p$, иначе он равен -1.

Определение 3. Символ Якоби. Пусть a и b – целые, взаимно-простые числа. Символ Якоби $J(a,b)$ равен $\prod_{p/b} L(a,p)^{l_p}$, где $|b| = \prod_{p/b} p^{l_p}$ – каноническое разложение на простые множители p.

Определение 4. Последовательность Люка. Пусть P и Q – два целых, таких что $D = P^2 - 4 \cdot Q \neq 0$. Главная последовательность Люка, связанная с парой (P, Q), определяется следующими рекуррентными уравнениями:

$$U_0(P,Q) = 0, \; U_1(P,Q) = 1,$$

$$U_{n+2}(P,Q) = P \cdot U_{n+1}(P,Q) - Q \cdot U_n(P,Q).$$

Добавочная последовательность Люка, связанная с парой (P, Q), определяется следующими рекуррентными уравнениями:

$$V_0(P,Q) = 2, \; V_1(P,Q) = P,$$

$$V_{n+2}(P,Q) = P \cdot V_{n+1}(P,Q) - Q \cdot V_n(P,Q).$$

Число D называется дискриминантом последовательности Люка U_0.

Математический аппарат, основанный на последовательностях Люка, позволяет построить протоколы цифровой подписи, аналогичные RSA и алгоритмам класса Эль-Гамаля, а также протокол обмена ключами, аналогичный протоколу Диффи-Хелмана. В [11] и [12] приведены соответствующие алгоритмы, а также метод вычисления членов последовательностей Люка, не использующий возведение в степень. Однако там рассматривается частный случай последовательностей, когда Q=1. Для общего случая последовательностей приведенная ниже теорема позволяет вычислять значения членов последовательности Люка с использованием классического бинарного метода возведения в степень.

Теорема 1. Пусть U_0 и V_0 – последовательности Люка, связанные с парой (P, Q). Тогда

$$\begin{pmatrix} U_n & V_n \\ U_{n-1} & V_{n-1} \end{pmatrix} = M^{n-1} \begin{pmatrix} 1 & P \\ 0 & 2 \end{pmatrix}, \text{ где } M = \begin{pmatrix} P & Q \\ 1 & 0 \end{pmatrix}.$$

Рассмотрим известные на сегодняшний день методы проверки на простоту, использующие разло-

жение N±1. Самым старым и самым известным на сегодняшний день является метод, основанный на теореме Люка. Полное доказательство теоремы можно найти в [13], здесь же приведена лишь ее формулировка.

Теорема 2 (теорема Люка). Пусть N – целое, тогда N – простое тогда и только тогда, когда существует такое целое a, взаимно-простое с N, что

$a^{N-1} \equiv 1 (\mathrm{mod}\, N)$ и $a^{\frac{N-1}{m}} \neq 1 (\mathrm{mod}\, N)$ для любого m – делителя N-1.

Чтобы воспользоваться этой теоремой для проверки на простоту, необходимо полное разложение N-1, что является вычислительно сложной задачей. В [14] доказана усовершенствованная теорема, использующая частичное разложение N-1, которая широко применяется.

Другая группа теорем для доказательства простоты базируется на разложении N+1, чтобы доказать простоту N. Данная теорема доказана Лемером в [15].

Теорема 3. Пусть N – целое, тогда N – простое тогда и только тогда, когда существует такая последовательность Люка U_m, дискриминант D которой $J(D, N) = -1$, и $U_{N+1} \equiv 0 (\mathrm{mod}\, N)$, $U_{(N+1)/R_i} \neq 0 (\mathrm{mod}\, N)$ для всех R_i – простых делителей N+1. Как и для предыдущего случая в [14], доказана теорема, использующая частичное разложение.

В [14] показано, что для доказательства простоты можно использовать частичное разложение N-1 и N+1. Это позволяет наиболее полно воспользоваться предположениями метода и уменьшить вероятность того, что известное частичное разложение недостаточно для доказательства простоты. В [14] доказана следующая теорема.

Теорема 4. Пусть N – нечетное целое, F1 – полностью разложенная на множители часть N-1 и F2 – полностью разложенная на множители часть N+1. Обозначим $\overline{F_2} = F_2 / 2$, $R_1 = (N-1)/F_1$ и $R_2 = (N+1)/F_2$. Предположим, что $НОД(R_1, F_1) = НОД(R_2, F_2) = 1$. Далее предположим, что:

1) для любого q_i – простого делителя F1 существует целое a_i, такое что

$$a_i^{N-1} \equiv 1 (\mathrm{mod}\, N),\ НОД(a_i^{(N-1)/q_i} - 1, N) = 1;$$

2) существует целое a, такое что

$$a_i^{N-1} \equiv 1 (\mathrm{mod}\, N),\ НОД(a_i^{(N-1)/R_1} - 1, N) = 1;$$

3) для каждого r_i – простого делителя F2 существует последовательность Люка $\left(U_k^i\right)$, дискриминант D‘ которой (дискриминант общий для всех последовательностей) такой, что $J(D, N) = -1$ и $N \setminus U^i_{N+1}, (U^i_{(N+1)/r_i}, N) = 1$ (где символ \ – делит);

4) существует последовательность Люка $\left(U_k^i\right)$, дискриминант D‘ которой такой, что $J(D, N) = -1$ и $N \setminus U^i_{N+1}, (U^i_{(N+1)/R2}, N) = 1$.

Далее предположим, что все простые делители R_1 и R_2 больше B_1 и B_2 соответственно. Определим r и s так, что $R_1 = \overline{F_2} \cdot s + r$. Пусть

$$G = \max\left(B_1 \cdot F_1 + 1, B_2 \cdot F_2 - 1, m \cdot F_1 \cdot \overline{F_2} + r \cdot F_1 + 1\right),$$

где $m \geq 1$. Далее, если $G = m \cdot F_1 \cdot \overline{F_2} + r \cdot F_1 + 1$, мы предполагаем, что $\left(\lambda \cdot F_1 \cdot \overline{F_2} + r \cdot F_1 + 1\right)$ не делит N для всех целых λ, $\delta_0^r \leq \lambda < m$, где δ_0^r – символ Кронекера. Выбирая m=1, мы исключаем все целые λ и таким образом упрощаем вычисления. Тогда если $N < G \cdot (B_1 \cdot B_2 \cdot F_1 \cdot F_2 + 1)$, то N – простое.

Как видно, первые два условия теоремы проверяют делители N-1, вторые два – делители N+1. Далее проверяется, известно ли достаточное количество делителей, чтобы утверждать, что если первые четыре условия верны, то N – простое. Грубо это обозначает, что с помощью теоремы 4 можно установить простоту числа N, если произведения факторизованных частей N-1 и N+1 превышают квадратный корень из N. Таким образом, с помощью теоремы 4 можно сертифицировать числа с известной значительной частью разложения N±1.

Далее нас интересует множество чисел N, таких что N-1 и N+1 имеют определенные большие простые делители. Пусть A (соответственно B) – нечетный делитель N-1 (соответственно N+1), тогда $N \equiv 1 (\mathrm{mod}\, A)$ (соответственно $N \equiv -1 (\mathrm{mod}\, B)$). Используется теорема Безу [13].

Теорема 5 (теорема Безу). Пусть A и B – два положительных, взаимно-простых, не равных нулю целых числа. Тогда существуют два целых числа u и v, такие что A·u - B·v = 1.

Согласно введенным обозначениям A и B всегда взаимно-простые. Действительно если d – общий делитель A и B, то он делит и N-1, и N+1, следовательно, и их разность, т.е. 2. Но d нечетный, так как A и B нечетные, таким образом, d=1.

Поскольку N – кандидат в простые числа, то N-1 и N+1 четные. Если N-1 делится на 2^k, при $k \geq 2$, N-1 делится на 4, а N+1 — нет. Наоборот, если N+1 делится на 2^k, то максимальная степень 2, которая делит N-1 – 2.

Следовательно, мы имеем два вида систем сравнений, определяющие множество искомых чисел:

$$\begin{cases} N \equiv 1 (\mathrm{mod}\, 2^k \cdot A), \\ N \equiv -1 (\mathrm{mod}\, B); \end{cases} \quad (2)$$

$$\begin{cases} N \equiv 1 (\mathrm{mod}\, A) \\ N \equiv -1 (\mathrm{mod}\, 2^k \cdot B). \end{cases} \quad (3)$$

Решения данных систем сравнений базируются на следующих теоремах.

Теорема 6. Пусть u и v — целые, такие что Au - Bv = 1, и пусть r и s — целые, такие что $2^k r$ - Bs = 1. Тогда N — решение системы (2) тогда и только тогда, когда $N \equiv 1 - u \cdot r \cdot 2^{k+1} \cdot A (\mathrm{mod}\, 2^k \cdot A \cdot B) \cdot$

Теорема 7. Пусть u и v — целые, такие что Bv - Au = 1, и пусть r и s — целые, такие что $2^k r$ - As = 1. Тогда N —решение системы (3) тогда и только тогда, когда $N \equiv -1 + v \cdot r \cdot 2^{k+1} \cdot B (\mathrm{mod}\, 2^k \cdot A \cdot B)$.

Следует отметить различие в формулах для построения u и v в теоремах 6 и 7. В [7] формула для u и v одинакова для обеих теорем: Au - Bv = 1. Однако в случае применения ее в теореме 7 мы вместо решения системы (3) получаем решение следующей системы:

$$\begin{cases} N \equiv -3 (\mathrm{mod}\, A) \\ N \equiv -1 (\mathrm{mod}\, 2^k \cdot B) \end{cases}, \quad (4)$$

Как видно из (4), в этом случае не может быть гарантировано заданное разложение N-1, т.е. предположения метода не выполнены.

По существу теоремы 6 и 7 задают два вида арифметических прогрессий, первый член и множитель которых взаимно-простые. Доказано, что такая прогрессия содержит простое число и, как следствие, бесконечное число простых чисел. Согласно результатам Линника [16] первое простое число в прогрессии a+k·d, где a и d взаимно-простые, оценивается сверху величиной d^{17}. Практические исследования показали, что первое простое число встречается гораздо раньше.

Приведенный математический аппарат позволяет построить все множество N с заданным разложением N±1 и проверить эти числа на простоту аналитически. При этом гарантирован факт, что множество чисел с заданным разложением содержит бесконечное число простых чисел.

2. Разработка алгоритмов построения простых чисел

На основании теоретических соотношений, изложенных в предыдущем разделе, предлагается алгоритм построения простого числа N с заданным разложением. Выбор последовательностей Люка для использования в проверке условий теоремы 4 будет осуществляться по правилу Бейли [6]: D— первое число из последовательности 5, 9, 13, 17... такой, что $J(D, N) = -1$; P — наименьшее нечетное число, большее $\sqrt{D}$. Если это значение не подходит, то увеличиваем P на 2. После нескольких неудачных попыток N — составное.

Исходными данными для алгоритма являются:

1) делители для N-1 — p_i, $i = \overline{1, l_1}$, $l_1 \geq 1$;

2) делители для N+1 — q_j, $j = \overline{1, l_2}$ q_j, $j = \overline{1, l_2}$, $l_2 \geq 1$;

3) type — тип используемой системы для построения N: type=0 — используется система (2), type=1 — используется система (3);

4) k — степень 2, входящая в разложение N-1 или N+1 в зависимости от значения type, $к \geq 2$;

5) len — требуемая битовая длина для получаемого числа N;

6) diping — число выбираемых по правилу Бейли последовательностей Люка для тестирования делителей N+1 перед отвержением числа как непростого, $diping \geq 2$.

После задания всех необходимых параметров поиск простого числа осуществляется по следующему алгоритму.

Алгоритм 1. Генерация простых чисел с заданным разложением.

1. Вычисляем: $A = \prod_{i=1}^{l_1} p_i$, $B = \prod_{i=1}^{l_2} q_i$. Проверяем: $НОД(A, B) = 1$, если не выполняется, то параметры заданы некорректно (см. предыдущий раздел). Происходит останов алгоритма.

2. Вычисляем:

u,v: $A \cdot u - B \cdot v = 1$, r, s: $2^k \cdot r - B \cdot s = 1$
(из теоремы 6), если type=0, или

$B \cdot v - A \cdot u = 1$, r, s: $2^k \cdot r - A \cdot s = 1$
(из теоремы 7), если type=1.

3. Вычисляем модуль $m = 2^k \cdot A \cdot B$.

4. Вычисляем: p_0: $p_0 \equiv 1 - u \cdot r \cdot 2^{k+1} \cdot A (\mathrm{mod}\, m)$, (из теоремы 6), если type=0, или

$p_0 \equiv -1 + v \cdot r \cdot 2^{k+1} \cdot B (\mathrm{mod}\, m)$ (из теоремы 7), если type=1.

5. Вычисляем: $N = p_0 + d \cdot m$, $d \geq 0$. Мы можем начинать поиск с любого члена арифметической прогрессии, определяемой теоремой 6 или 7. N — первое число, которое будет проверяться на простоту.

6. Проверяем N на простоту по алгоритму 2.

7. Если N — простое, N — искомое число, если нет, N = N + m и перейти к шагу 6. Если недостаточно делителей для сертификации числа по алгоритму 2, то происходит останов алгоритма с выдачей сообщения о недостаточных исходных данных.

Этот алгоритм последовательно просматривает все числа натурального ряда, имеющие разложение N±1, задаваемое исходными данными. Останов алгоритма происходит в случае полной сертификации числа по алгоритму 2 или если алгоритм 2 покажет, что задано недостаточное количество делителей для полной сертификации числа. В предыдущем разделе было показано, что такое число обязательно встретится.

Проверка на простоту числа с заданным разложением осуществляется по теореме 4.

Алгоритм 2. Проверка на простоту.

1. Выбираем a=2. Если соотношение $a^{N-1} \equiv 1 (\mathrm{mod}\, N)$ не выполняется, то N — составное (по теореме 2).

2. Вычисляем: $F_1 = 2^k \cdot A$, $F_2 = 2 \cdot B$, если type=0, или $F_1 = 2 \cdot A$ $F_2 = 2^k \cdot B$, если type=1, $R_1 = N / F_1$, $R_2 = N / F_2$.

3. Проверяем: $НОД(R_1, F_1) = НОД(R_2, F_2) = 1$. Если не выполняется, то N – составное.

4. Выбираем a из ряда натуральных чисел, начиная с 3, такое, что $j = J(a_i, N) = -1$. Вычисляем $t = a^{(N-1)/2} (\mathrm{mod}\, N)$. Если $t \neq j$, то N – составное (из теста Соловея-Штрассена [4]), иначе первое условие теоремы 4 для 2, как делителя N-1, выполнено.

5. Выбираем a из ряда натуральных чисел, начиная с 3, такое, что $j = J(a_i, N) = 1$. Вычисляем $t = a^{(N-1)/2} (\mathrm{mod}\, N)$. Если $t \neq j$, то N – составное. Проверяем:

$a^{N-1} \equiv 1 (\mathrm{mod}\, N)$, $НОД(a^{(N-1)/p_i} - 1, N) = 1$, $i = \overline{1, l_1}$,

$НОД(a^{(N-1)/R1} - 1, N) = 1$. Если не выполняется, N – составное, иначе условия теормы 4 для нечетных делителей выполнено.

6. Пользуясь правилом Бейли, выбираем последовательности Люка $\left(U_k^i\right)$ и проверяем

$N \setminus U^i_{N+1}, (U^i_{(N+1)/q_i}, N) = 1, i = \overline{1, l_2}$ и

$N \setminus U^i_{N+1}, (U^i_{(N+1)/R2}, N) = 1$. Если перебрав diping последовательностей, не находим нужной, то N – составное.

7. Присваиваем B_1=2, B_2=2. Это позволяет не раскладывать R_1 и R_2 на множители для проверки последнего условия теоремы 4.

8. Находим G и проверяем последнее условие теоремы 4. Если не выполняется, то N – составное.

Если число N успешно прошло все этапы алгоритма 2, то это число – простое, а следовательно, N – результат работы алгоритма. Следует отметить, что в большинстве случаев отвержение составного числа происходит на шаге 1 алгоритма 2. Небольшая часть чисел отвергается на шаге 3. Эксперименты показали, что числа, подлежащие отбраковке по пункту 3, в основном проходят сертификацию по алгоритму 2 и во всех исследованных случаях $НОД(R_1, F_1) = 2^m$, если type=0, или $НОД(R_2, F_2) = 2^m, m \geq 1$, если type=1. Этот факт объясняется тем, что по условиям систем (2) и (3) в разложение N+1 и N-1 соответственно входит 2^1, поэтому $2^m, m \geq 2$ не может входить в разложение указанных чисел по условию.

Если число N не удовлетворяет условию пункта 3 алгоритма 3, его разложение N не будет в точности соответствовать заданному в исходных данных – реальное к будет больше задаваемого в исходных данных. Решение об отвержении чисел было принято в целях построения чисел, точно соответствующих параметрам, задаваемым в исходных данных.

Во время проверки числа на соответствие условиям 3 и 4 теоремы 4 (пункт 6 алгоритма 2) перебирается diping различных последовательностей Люка. Аналогично проверке N на соответствие первым двум пунктам теоремы 4, где число a_i для нечетных делителей N-1 и для 2 различно, для нечетных делителей N+1 и для 2, чтобы проверить условия 3 и 4, требуются различные последовательности Люка. Таким образом, diping должно быть ≥ 2. Если diping будет равно 1, то ни одно число не пройдет проверку. Желательно назначать diping равным 3, 4, и более, чтобы исключить отбраковку алгоритмом простых чисел.

Проверка последнего условия теоремы 4 определяет, достаточно ли делителей N±1 задано. В принципе при необходимости можно считать, что уже получены определенные гарантии простоты числа, не прошедшего последнюю проверку, однако утверждать, что число действительно простое, нельзя, поэтому оно не может быть результатом работы аналитического метода проверки чисел на простоту. Для построения истинно простого числа необходимо дополнить исходные данные.

Таким образом, работа метода состоит из двух этапов – поиска простого числа по алгоритму 1 и его полной сертификации по алгоритму 2. Далее будут рассмотрены особенности реализации и временные характеристики метода.

3. Реализация алгоритма

Данный алгоритм был реализован на алгоритмическом языке Borland C++ 3.1. Для выполнения операций многократной точности использовалась библиотека многократной арифметики MMATH v 1.12, разработанная АО ИИТ (г. Харьков), реализующая блочный метод Кнута для возведения чисел многократной точности в степень по модулю. Поскольку часть используемого в методе математического аппарата не является широко используемой, ряд подпрограмм, таких как, например, вычисление чисел, удовлетворяющих теореме Безу (теорема 5), вычисление символа Якоби, а также нахождение значения произвольного члена последовательности Люка, разрабатывались специально для реализации данного метода.

Все вспомогательные подпрограммы, а также функция GenPrime, непосредственно реализующая алгоритмы 1 и 2, помещены в отдельный модуль, который может быть откомпилирован в составе проектного файла Borland C++ 3.1, что позволяет использовать метод для генерации простых чисел непосредственно из приложений, разрабатываемых на этом языке программирования.

Одна из разработанных подпрограмм позволяет на основании случайно выбираемых из таблицы простых числе длиной 10 бит получать случайные простые числа длиной 32 бита и более. В основу данной подпрограммы положен эвристический метод постепенного увеличения длины числа, вызывающий GenPrime несколько раз. Получаемые простые числа не могут считаться сильными, однако их простота гарантирована, а следовательно, их можно использовать в качестве параметров для генерации сильных простых чисел.

В случае необходимости функция GenPrime может выводить на стандартное устройство вывода информацию о процессе поиска и сертификации числа, а также временные характеристики.

Для демонстрации работы метода и исследования его статистических и временных характеристик было разработано DOS-приложение gpdemo. Оно позволяет получать простые числа нужного вида с возможностью их сохранения в десятичном виде в

текстовом файле и в двоичном виде — в бинарном. Кроме того, создается файл с временными характеристиками процесса генерации чисел. Из текстового файла gpdemo получает информацию о количестве требуемых чисел, их длине, исходных характеристиках и длинах делителей. Конкретные делители нужной длины подбираются автоматически. Результаты экспериментов, проведенных с этим приложением, описаны в следующем разделе.

Для обеспечения более дружественного интерфейса и удобства проведения исследований было разработано Windows'95-приложение GPWin. Разработка велась на алгоритмическом языке Borland Delphi 3.0, предоставляющем широкие возможности для разработки дружественных интерфейсов.

GPWin осуществляет редактирование исходного файла для gpdemo и производит анализ результатов. Имеется возможность загружать и сохранять файлы конфигурации для gpdemo, одновременно являющиеся файлами конфигурации для GPWin. Приложение снабжено контекстно-зависимой помощью.

GPWin состоит из двух окон – окна параметров, позволяющего загружать и редактировать параметры генератора простых чисел, и окна результатов, показывающего результирующие числа и производящего статистический анализ.

Имеется возможность задания всех необходимых параметров для алгоритма генерации, а также дополнительная возможность рандомизации чисел с помощью параметра "степень разброса", определяющего, насколько сильно полученные числа будут распределены по участку натурального ряда для заданной битовой длины. Все установки можно загрузить из файла или сохранить в файле. После задания всех параметров можно приступить к генерации чисел. Через некоторое время итоги работы будут показаны в окне результатов.

В этом окне можно просмотреть все числа, полученные в результате работы метода, вместе со временем их генерации, а также проанализировать распределение этого времени по интервалу между максимальным и минимальным, увидеть среднее время генерации и его дисперсию. Это позволяет выдвинуть и проверить гипотезу о законе распределения времени генерации простого числа с заданным разложением, а значит, как следует из теоретических сведений и алгоритма метода, и о вероятности появления простого числа в арифметической прогрессии при заданной битовой длине.

Таким образом, разработанные приложения позволяют полностью исследовать характеристики работы генератора простых чисел, а также закономерности, связанные с простыми числами.

4. Результаты экспериментальных исследований

Тестирование метода осуществлялось при помощи программных средств, описанных выше. Прежде всего, интересовали временные характеристики метода, так как длительное время генерации – основной недостаток аналитических методов проверки на простоту.

Для построения временных характеристик метода генерировалось по 100 чисел длиной 128, 256, 512 и 1024 бита. N-1 и N+1 имели по 2 делителя длиной, близкой к 1/4 требуемой длины числа. Общая длина делителей ограничена не только снизу необходимостью того, чтобы их было достаточно для сертификации числа, но и сверху, если мы хотим получить число заданной длины. Это следует из теорем 6 и 7. Из формул для арифметических прогрессий, определяемых этими теоремами, видно, что если величина модуля $2^k \cdot A \cdot B$ будет слишком большой, то число N быстро превысит границу битовой длины.

При тестировании использовались системы типа (2) и (3). Эксперименты показали, что временные характеристики не зависят от типа системы. К этому же выводу можно прийти, анализируя алгоритмы 1 и 2. В зависимости от типа системы только в пункте 4 алгоритма 1 производятся различные вычисления, однако эквивалентные по сложности. К тому же эти вычисления производятся один раз за сеанс генерации простого числа.

Используемая степень двойки k принималась константой $k = 4$. Как видно из алгоритма 1, k участвует только в формировании шага арифметической прогрессии.

Для каждой битовой длины измерялось полное время генерации и время прохождения сертификации по алгоритму 2. Практические исследования показали, что большинство тестируемых составных чисел отвергаются по пункту 1 алгоритма 2 – возведение числа 2 в степень N-1 по модулю N. Это означает, что практически отвержение непростого члена прогрессии по вычислительным затратам равно одному возведению в степень по модулю числа заданной битовой длины. Полученное минимальное и среднее время полной генерации, а также минимальное и максимальное время полной сертификации для чисел различной длины приведено в таблице.

L	T1	T2	T3	T4
128	0,17	0,67	0,11	0,33
256	0,71	5,40	0,61	3,5
512	2,80	31,9	2,80	31,47
1024	70,34	550,76	40,27	64,38

L – размер числа, бит; T1 – время генерации, минимальное, с.; T2 – время генерации, среднее, с.; T3 – время сертификации, минимальное, с.; T4 – время сертификации, максимальное, с.

Испытания проводились на ПК Pentium 133, работающем под управлением операционной системы MS-DOS.

Графически зависимость среднего и минимального времени полной генерации числа от заданной битовой длины представлена на рис. 1, а минимальное и максимальное время сертификации – на рис. 2.

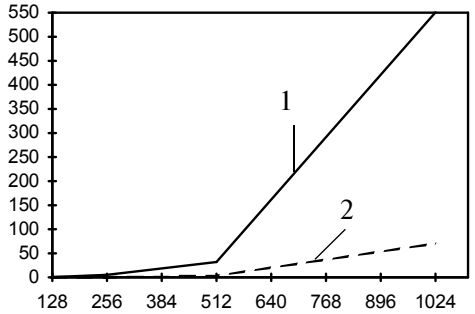

Рис. 1. Зависимость минимального (2) и среднего (1) времени генерации от битовой длины числа

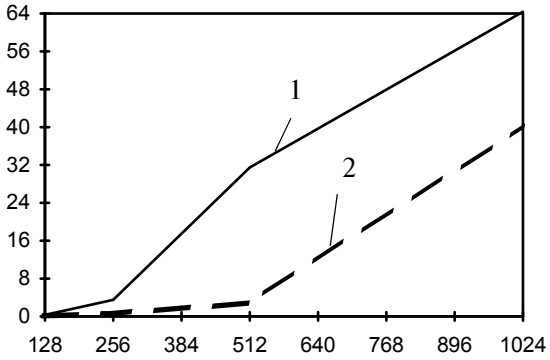

Рис. 2. Зависимость минимального (2) и максимального (1) времени сертификации от битовой длины числа

Время полной сертификации числа хотя и различно, однако при заданной его длине принимает несколько фиксированных значений, поэтому основную неопределенность относительно времени генерации числа составляет время поиска. Эксперименты показали, что время полной генерации числа, возможно, подчиняется экспоненциальному закону распределения. Это видно из гистограммы на рис. 3, построенной для времени полной генерации 100 чисел длиной 512 бит.

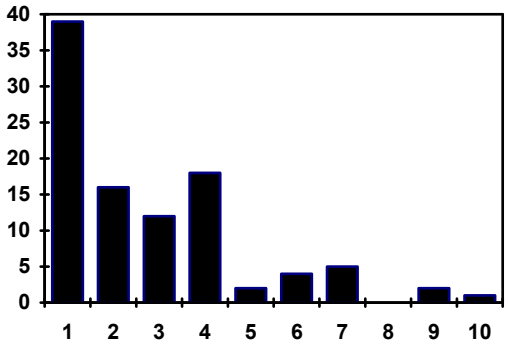

Рис. 3. Гистограмма распределения времени генерации 100 простых чисел длиной 512 бит

Экспоненциальный закон распределения времени полной генерации числа означает, что вероятнее всего простое число обнаружится намного раньше среднего времени полной генерации.

Каждое полученное число проверялось 100 циклами теста Рабина-Миллера [5], подтвердившими их простоту. Это еще раз показывает, что метод действительно позволяет генерировать простые числа, причем эти числа при правильном подборе параметров будут гораздо более стойкими к известным методам факторизации N±1, чем псевдопростые числа.

Заключение

Полученные временные характеристики показывают, что данный метод вполне применим при построении простых чисел для долговременных ключей, время генерации которых не критично. Так, чтобы сгенерировать два сильных 512-битных числа P и Q для образования 1024-битного модуля N для RSA-криптосистемы, потребуется в среднем от 30 секунд до одной минуты. Учитывая, что модуль является долговременным параметром, это время вполне приемлемо даже по сравнению с вероятностными тестами, которые работают на порядок быстрее, но не дают гарантию простоты числа и не позволяют получать сильные простые числа.

Поскольку предложенный выше метод генерации простых чисел аналитический, при соблюдении всех требований получаемое на выходе число всегда простое, к тому же оно специального вида. Разумеется, компрометация задаваемых в исходных данных делителей сделает легко решаемой задачу факторизации полученного модуля N для RSA системы. Поэтому при использовании данного метода необходимо строгое соблюдение требований конфиденциальности входных параметров. Выполнение этой задачи возлагается на пользователя методом. В то же время, получая возможность определять вид числа, пользователь может быть уверен в его свойствах.

Аналитические методы, основанные на знании полного разложения N-1 или N+1, вычислительно сложны и практически не пригодны для реализации. В алгоритме реализации ГОСТ Р 3410-94 при генерации общего модуля для цифровых подписей класса Эль-Гамаля применяется итерационный метод, основанный на постепенном удвоении длины числа до нужной, начиная с минимального простого 32-битного. Простое число, полученное на i-м шаге, используется в качестве делителя N-1 для числа, получаемого на i+1-м шаге. Это дает возможность использовать малую теорему Ферма для проверки числа на простоту. Этот метод не позволяет контролировать разложение N+1 получаемых чисел. В описываемом методе можно контролировать оба разложения и при этом обеспечивать случайность выбора числа из множества простых натуральных чисел заданного разложения и заданной длины.

При реализации метода используется математический аппарат, нетрадиционный для реализации подобных методов. Базовая теорема метода объединяет две теоремы доказательства простоты, основанные на частичном разложении N-1 и N+1. Методы, основанные на разложении N+1, получили меньшее распространение из-за кажущейся большой вычислительной сложности, но в настоящее время предпринимаются попытки создания алгоритмов, не уступающих по скорости возведению числа в степень по модулю, основанные на использовании частных случаев последовательностей. На этом примере видно, как объединение двух различных подходов к проблеме нахождения простого числа может привести к созданию эффективного метода.

Литература: 1. *R.L. Rivest, A. Shamir, L.M. Adleman.*– A method for obtaining digital signatures and public key cryptosystems. Comm. ACM, 1978. 300 p. 2. *T. El Gamal.* A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions of Information Theory (July 1985). P. 150-155. 3. *W. Diffie and M. E. Hellman.* New directions in cryptography. IEEE Transactions of Information Theory (November 1976). P. 300-307. 4. *R. Solovay and V. Strassen.* A fast Monte Carlo method for primality. SIAM J. Comput., 1977. P. 50-58. 5. *G.L. Miller.* Riemann's hypothesis and tests for primality. J. comp. Syst. Sci., 13, 1976. P. 83-89. 6. *R. Baillie and S. S. Wagstaff.*–Lucas pseudoprimes. Marh. Comp., 35, 1980. P. 120-129. 7. Yannick Saouter A new method for the generation of strong prime numbers. IRISA, 1995. 16 p.

8. *J. Gordon* Strong primes are easy to find. – In LNCS 200. Eurocrypt 84, 1984. P.111-115. 9. *J.M. Pollard.* A montecarlo method for factorization. BIT, 15. 1975. P. 331-334. 10. *Lenstra H.W., Jr.* Primality testing algorithms (after Adleman, Rumely and Williams), Sem. Bourbaki, 1980/1981. Vol. 33. P. 67-73. 11. *Smith P.* LUC public key encryption. Dr. Dobb's Journal (January 1993). P. 100-108. 12. *Smith P.* Cryptography without exponentation. Dr. DOB's Journal (April 1996). P. 120-123. 13. *Виноградов И. М.* Основы теории чисел. М.: Наука, 1981. 300 с. 14. J. BrillHard, D. H. Lehmer, and J. L. Selfridge. New primality criteria and factorizations of $2^m \pm 1$. Math. Comp., 1975. 29. P. 50-58. 15. *D. H. Lehmer.* An extended theory of Lucas functions. Annals of Math., 31. 1930. P. 419-448. 16. *Y. V. Linnik.* On the least prime in the aritjmetic progression i. The basic theorem. Mat. Sbornik 15(57) 1944. P. 111-117.

Поступила в редколлегию 12.06.98

Рецензент: д-р техн. наук, проф. Долгов В.И.

Горбенко Иван Дмитриевич, д-р техн. наук., профессор, проректор по научной работе ХТУРЭ. Научные интересы: защита информации в компьютерных системах и сетях. Адрес: 310726, Украина, Харьков, пр. Ленина, 14, тел. 30-24-50, 37-56-39.

Жилин Олег Витальевич, студент ХТУРЭ. Научные интересы: защита информации в компьютерных системах и сетях. Адрес: 310064, Украина, Харьков-64, Комсомольское шоссе, 49 а, кв.33.

УДК 519.7

ЧАСТОТНО-ИМПУЛЬСНОЕ КОДИРОВАНИЕ ЗВУКОВЫХ СИГНАЛОВ

БАРДАНОВ Е.Б., КОРЯК С.Ф., МАЛЕНЧЕНКО З.Ю., ШАБАНОВ-КУШНАРЕНКО Ю.П.

Изложена теория частотно-импульсного кодирования звуковых сигналов, обеспечивающего идентичность слухового восприятия натурального и кодированного звуков. Усилен обощенный закон Талбота, экспериментально определены численные значения параметров кода звука, обеспечивающие натуральность звучания. Установлена линейность зависимости критического тактового интервала от минимального интервала между соседними импульсами синхронизированного частотно-импульсного кода звука.

Введение

Неоценимым подспорьем при решении задач автоматической обработки акустических сигналов может служить изучение механизмов слухового восприятия этих же сигналов человеком. Ведь сами-то люди отлично справляются со всеми задачами обработки акустических сигналов, которые желательно решать с помощью акустической аппаратуры. Если такое изучение удастся выполнить в достаточно полном объеме, то будет открыт путь к созданию автоматически действующих приборов и систем, эффективно обрабатывающих разнообразные акустические сигналы. Одна из первоочередных задач в этой области заключается в том, чтобы выяснить те принципы и математически описать закономерности, на основе которых ухо воспринимает непрерывные колебания воздуха и превращает их в дискретную форму.

В большинстве современных приборов и систем приема, хранения и переработки акустических сигналов первым преобразованием электрического сигнала, поступающего с выхода микрофона, является его дискретизация. В этом случае принято использовать аналого-цифровой преобразователь [1]. Обычно формируется 8-12 — разрядный двоичный код ординаты решетчатой функции акустического колебания, что соответствует ее измерению примерно с двумя-тремя знаками точности. Интервал между соседними точками принимается, как правило, в пределах от 100 мкс до 1 мс. Акустическое воспроизведение подобного кода не обеспечивает идентичности его восприятия с исходным звуком. В ряде случаев используют гораздо более точные коды. Например, в работе [2] описано применение 16-разрядных двоичных кодов при частоте дискретизации 68 кГц. Синтезированный по такому коду звук мало отличается на слух от исходного. Широкое практическое использование дискретизации акустических сигналов выдвигает для научной проработки ряд актуальных вопросов: 1) можно ли строго обосновать правомерность дискретизации звука; 2) является ли использование аналого-цифрового преобразователя стандартного типа наилучшим способом дискретизации звукового сигнала; 3) с какой точностью и частотой следует проводить дискретизацию и на чем основывать выбор численных значений параметров дискретизации. Анализ литературных данных показывает, что ни на один из этих вопросов пока нет исчерпывающих ответов.

Обычно для обоснования правомерности дискретизации речевого сигнала ссылаются на теорему дискретизации [3], которая утверждает, что когда спектр $J(\omega)$ функции $f(t)$, заданной на всей вещественной оси, тождественно равен нулю при частотах $\omega \geq \omega_0$, то по точным значениям функции $f(t)$, найденным в точках $t=0, \pm 2\pi/\omega_0, \pm(2\pi/\omega_0)2, \ldots$, можно полностью восстановить вид функции $f(t)$. Основываясь на этой теореме, часто делают следующий вывод: поскольку ухо не слышит звуков при частотах выше 20 кГц, то для получения полноценного кода звука можно воспользоваться его цифровым кодом, полученным при удвоенной частоте дискретизации, равной 40 кГц [5].

Этот вывод представляется не вполне обоснованным. Во-первых, в теореме дискретизации говорится о вещественных, т.е. об идеально точных значениях функции $f(t)$. На вопрос о том, можно ли заменить эти значения их конечными цифровыми кодами и сколько знаков надо взять в этих кодах, она вообще не дает какого-либо ответа. Между тем ясно, что идеально точно измерить значения функции $f(t)$ при ее дискретизации практически невозможно. Во-вторых, из того факта, что ухо не слышит звуков на частотах выше 20 кГц, вовсе не следует, что в звуках, воспринимаемых слуховым анализатором, нет частот выше 20 кГц. Напротив, точно известно, что они там имеются. Более того, установлено, что часть из этих частот существенно влияет на слуховое восприятие звука. Так, известно, что исключение из акустического колебания музыкального произведения частот в диапазоне 20-40 кГц заметно обедняет звучание этого произведения. В-третьих, и это главное,