УДК 62-50:519.7

[1]**Кириченко В.В.,** к.ф.-м.н.,
[2]**Лесина Е.В.,** к.ф.-м.н.

ОБ ОСОБЕННОСТЯХ ПРИМЕНЕНИЯ ДИНАМИЧЕСКИХ СИСТЕМ В АЛГОРИТМАХ ЗАЩИТЫ ДАННЫХ

[1]Национальный авиационный университет
[2]Донецкий национальный технический университет

vkir28@gmail.com
lesina17@gmail.com

Представлены некоторые сведения о новых методах в задачах обработки информации, основанных на использовании теории динамических систем и, в частности, теории динамического хаоса. Рассматриваются особенности построения криптографических алгоритмов, основанных на нелинейных дискретных динамических системах. Предложена математическая модель такого алгоритма, а также способы проверки эффективности его работы. Рассмотрены особенности реализации операций в конечном кольце целых чисел в микропроцессорной системе

Ключевые слова: конечный автомат, обратимые динамические системы, конечномерное кольцо целых чисел, генераторы псевдослучайных последовательностей

Введение

Вследствие лавинообразного распространения компьютерных систем и их взаимодействия посредством сетей наблюдается все большая зависимость, как организаций, так и отдельных людей от информации, пересылаемой по сети и хранящейся в таких системах. Это, в свою очередь, заставляет осознать необходимость защиты данных и ресурсов от возможности несанкционированного доступа, важность использования специальных средств для обеспечения достоверности получаемых данных и сообщений, а также защиты систем от сетевых атак.

Динамические системы, обладающие хаотичным поведением, в настоящее время интенсивно используются и применяются в различных областях, в частности, для криптографической защиты информации [1,2]. На основе таких систем могут быть построены генераторы псевдослучайных последовательностей, которые в дальнейшем используются для кодирования открытого текста [3, 4]. С другой стороны, всякая динамическая система, имеющая структуру вход-выход, может использоваться непосредственно для преобразования данных. На основе таких систем создается шифратор. Входом в систему служит оцифрованное сообщение, а выходом является зашифрованный сигнал, направленный в телекоммуникационные сети. Необходимым условием для однозначной дешифровки является существование обратной системы [5].

Автоматные аналоги динамических хаотических систем

Вычисления с ограниченной точностью можно выполнить, переходя от дифференциальных уравнений к конечно-разностным. Если в силу конечной разрядности любого компьютера ограничиваться конечным числом значений всех параметров и переменных, входящих в исходную систему, то полученную систему уравнений можно рассматривать как описание некоторого конечного автомата.

Конечный автомат понимается как пятерка объектов $A=(S,X,Y,d,l)$, где S – (конечное) множество состояний, X – (конечный) входной алфавит, Y – (конечный) выходной алфавит,

$d: S\times X \to S$ – функция переходов,

$l: S\times X \to Y$ – функция выходов.

Переход от дифференциальных уравнений к их конечно-разностным аналогам приводит к уравнениям, в которых

присутствуют 4 арифметических операции. Конечность числа значений, участвующих в них величин (в силу вышеприведенных соображений) и необходимость сохранения формы уравнений, отражающей связи между этими величинами, делают естественным рассмотрение этих уравнений как уравнений в конечных полях (или, в более простых ситуациях, в конечных кольцах).

Порядком поля называется число элементов в нем. Поле порядка q обозначается через $GF(q)$. Если диапазон изменения величин, встречающихся в упомянутых выше конечно-разностных уравнениях ограничен и конечен, то можно подобрать подходящий порядок q поля, при котором указанные величины будут инъективно отображаться в элементы поля $GF(q)$, а интерпретация арифметических операций как операций в поле $GF(q)$ даст описание автомата соответствующей системой уравнений.

Рассмотрим работу такого преобразователя на примере системы n нелинейных дифференциальных уравнений первого порядка:

$$\dot{x}_k = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}^k x_i x_j + \sum_{i=1}^{n} B_i^k x_i , \qquad (1)$$

$k = \overline{(1,n)}$.

Здесь матрицы $A^k, k = \overline{1,n}$ имеют размерность $n \times n$ и определяют нелинейную часть системы. Векторы $B^k(b_1^k, \mathbf{K}, b_n^k)$, $k = \overline{1,n}$ - отвечают за ее линейные члены. Если все коэффициенты матриц $A^k, k = \overline{1,n}$ равны нулю, то система (1) становится линейной. Ее решение можно записать аналитически, а траектории имеют регулярный или периодический характер. Если хотя бы один из коэффициентов матриц $A^k, k = \overline{1,n}$ отличен от нуля, то в системе появляется нелинейность. Такие системы имеют как регулярные, так и стохастические траектории, что значительно усложняет, а то и делает невозможным их аналитическое исследование. Этот факт с учетом простоты структуры таких систем, широко используется для реализации различных систем

Из-за отсутствия точных методов решения нелинейных динамических систем общего вида для их анализа часто применяют численные методы – такие, как, например, сочетание явной схемы Эйлера с центрально-разностной схемой Адамса, использование старших производных, а также метода Рунге-Кутта 4-ого порядка. В случае классических значений параметров системы наблюдается неустойчивость ее решений, поскольку положения равновесия системы имеют седловой тип. Это ограничивает применение указанных методов, поскольку растет общая ошибка с увеличением отрезка интегрирования. Таким образом, небольшие изменения в начальных условиях системы (1) могут привести со временем к значительным последствиям. Это свойство системы позволяет ее использовать для надежной защиты информации.

Рассмотрим задачу преобразования данных с помощью системы (1). Для этого изменим $\hat{k}$-е уравнение (здесь $\hat{k}$ - фиксированное число $1 \le k \le n$), добавив в него входной сигнал $u(t)$, а выходным сигналом является $y(t)$, в результате чего $\hat{k}$-е уравнение приобретает вид:

$$\dot{x}_{\hat{k}} = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}^{\hat{k}} x_i x_j + \sum_{i=1}^{n} B_i^{\hat{k}} x_i + au .$$

Дискретизация с шагом h приводит к уравнениям, которые в модифицированном виде имеют вид:

$$x_k(t+1) = x_k(t) + \qquad (2)$$

$$+ h\left(\sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}^k x_i(t) x_j(t) + \sum_{i=1}^{n} B_i^k x_i(t) \right)$$

$k = (\overline{1, \hat{k}-1}, \overline{\hat{k}+1, n})$,

$$\left(x_k(t+1) - x_{\hat{k}}(t)\right)/h =$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}^{\hat{k}} x_i(t) x_j(t) + \sum_{i=1}^{n} B_i^{\hat{k}} x_i(t) + au(t) .$$

Выход $y(t)$ описывается уравнением

$$y(t) = x_{\hat{k}}(t+1) .$$

В обратной системе входной и выходной символы меняются местами, то есть входом обратной системы является $y(t)$, а выходом – $u(t)$. Она содержит те же уравнения при $k = (\overline{1, \hat{k}-1}, \overline{\hat{k}+1, n})$, а $\hat{k}$-е заменяется, соответственно, на следующие:

$$x_{\hat{k}}(t+1) = y(t)$$

$$u(t) = \frac{1}{a}\left(\frac{x_{\hat{k}}(t+1) - x_{\hat{k}}(t)}{h} - \right.$$

$$\left. - \sum_{i=1}^{n} \sum_{j=1}^{n} A_{i,j}^{\hat{k}} x_i(t) x_j(t) - \sum_{i=1}^{n} B_i^{\hat{k}} x_i(t) \right)$$

Автомат, описываемый уравнениями (2), назовем прямым автоматом $L(A, B, a, h)$, а соответствующий обратной системе – обратным автоматом $L^{-1}(A, B, a, h)$.

Так как обычно речь идет об обработке данных микроконтроллером, то они представляется последовательностью битов, более крупных единиц – байтов или кратных байтам. В этом случае число различных элементов, описываемых всевозможными комбинациями значений отдельных битов, равно $2^m = q$, где $m = 8k,\ k \in N$. Поэтому соответствующие вычисления можно проводить либо в кольце Z_q, либо в поле $GF(2^m)$.

Особенности реализации алгоритма шифрования

Для исследования свойств алгоритмов шифрования, использующих систему вида (2), авторами работы был реализован комплекс программ на языке программирования C++. Система (1) при этом использовалась со следующими ненулевыми параметрами: $A_{22}^1 = p_1$, $A_{13}^2 = q_1$, $A_{12}^3 = r_1$, $B_3^1 = p_2$, $B_1^2 = q_2$, $B_1^3 = r_2$, $B_2^3 = r_3$. В таком случае система (4) с шагом дискретизации $h = 1$ и с операциями в кольце целых чисел Z_m принимает вид

$$\begin{cases} x_1(t+1) = x_1(t) + \\ \qquad - x_2^2(t) + p \cdot x_3(t) \mod 2^m, \\ x_2(t+1) = x_2(t) + \\ \quad + x_1(t) x_3(t) - q \cdot x_1(t) + \\ \qquad + u(t) \mod 2^m, \\ x_3(t+1) == x_3(t) + r_1 \cdot x_1(t) + \\ \quad + r_2 \cdot x_2(t) - x_1(t) x_2(t) \mod 2^m. \end{cases} \quad (3)$$

Здесь $u(t)$ – входной сигнал, $x_2(t)$ - закодированный сигнал, который передается по каналу передачи данных. Параметры $key = (p, q, r_1, r_2)$ и начальные условия $x^{in} = \left(x_1(0), x_2(0), x_3(0)\right)$ системы, которые однозначно определяют выходной сигнал, образуют ключ системы шифрования. Таким образом прямой автомат $L(key, x^{in})$ определяет систему шифрования или систему-передатчик.

Система-приемник принимает зашифрованный сигнал и с помощью обратного автомата $L^{-1}(key, x^{in})$ декодирует его в исходный. То есть и для операции шифрования, и для операции дешифрования необходим один и тот же ключ. Даже незначительное нарушение его структуры хотя бы в одном узле приведет к невозможности правильного восстановления передаваемого сигнала.

Для восстановления сигнала необходимо использовать следующую систему:

$$\begin{cases} x_1(t+1) = x_1(t) - x_2^2(t) + \\ \qquad + p \cdot x_3(t) \mod 2^m, \\ u(t) = x_2(t+1) - x_2(t) - \\ \quad - x_1(t) x_3(t) + q \cdot x_1(t) \mod 2^m, \\ x_3(t+1) = x_3(t) + r_1 \cdot x_1(t) + \\ \quad + r_2 \cdot x_2(t) - x_1(t) x_2(t) \mod 2^m. \end{cases}$$

В результате проведенных тестов с использованием данного комплекса были выявлены следующие проблемы и особенности реализации алгоритмов шифрования.

Для демонстрации обычно алгоритм шифрования реализуется по одной из следующих схем.

Шифрование текстового массива происходит по следующей схеме:

Входной текст → Символ → ASCII код →

→ $\boxed{L(key, x^{in})}$ →

→ ASCII Код → Символ → Выходной текст.

Входным сигналом является текстовое сообщение. За одну итерацию считывается один символ из текста. Из таблицы текстовых кодов ASCII (или другой) берется соответствующий данному символу код, который прямым автоматом $L(key, x^{in})$ преобразуется в другое значение. Автомат при этом использует заданный ключ. Полученное значение с помощью таблицы символов преобразуется в текстовый символ. Таким образом формируется новое сообщение из текстовых символов, которое является зашифрованным сигналом. Расшифровка происходит по тому же принципу, только используется обратный автомат, а в качестве входного текста – зашифрованный сигнал. Далее приводятся некий исходный текст, зашифрованный и расшифрованный. Если последний с первым совпадают, то значит, что алгоритм работает корректно. Существенным недостатком такого способа является то, что по виду зашифрованного сигнала ничего нельзя сказать про эффективность работы такого алгоритма.

Шифрование числового массива. Для демонстрации работы алгоритма генерируется последовательность чисел специального вида, как правило описывающая некий аналоговый сигнал. После этого последовательность преобразуется с помощью прямого автомата:

Входной цифровой массив → Число →

→ $\boxed{L(key, x^{in})}$ →

→ Число → Выходной числовой массив.

В данном случае последовательности удобно отображать графически. На рисунке 1 показан график аналогово сигнала, имеющего периодический характер. А также зашифрованный сигнал с помощью прямого автомата, описываемого системой (3).

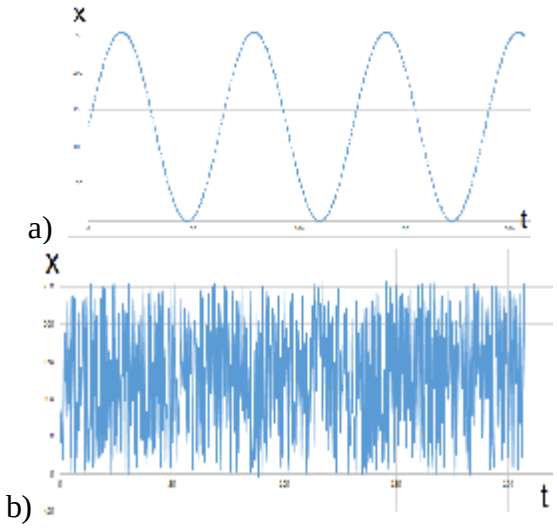


Рис. 1. Шифрование периодического сигнала
a) исходный b) зашифрованный.

Здесь t – время, в течении которого передается сигнал, x – его амплитуда. График на рисунке 1б показывает достаточною хаотичность зашифрованного сигнала. Таким образом можно сделать вывод, что предложенный алгоритм работает достаточно эффективно. Однако для более глубоко анализа необходимо проводить статистические тесты.

Шифрование бинарного потока. Работа с логическими схемами, которые используются как в телекоммуникацион-

ных, так и компьютерных системах, часто приводит к алфавиту {0, 1}. Поэтому актуальным является вопрос создания бинарной криптосистемы. В этом случае шифруется поток битов. На каждой итерации считывается последовательность k бит, которая затем переводится в десятичное число в интервале $(0,2^k)$. С помощью прямого автомата число преобразуется в другое, а то в свою очередь в двоичную последовательность битов. Таким образом получается новая бинарная последовательность зашифрованных данных:

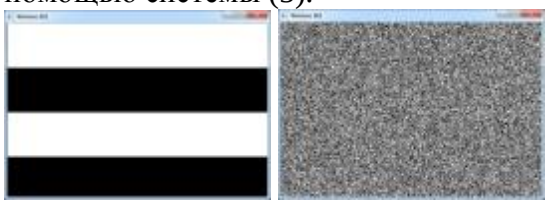
Входной бинарный поток → k бит →

→ Число $(0,2^k)$ →

→ $L(key, x^{in})$ → Число $(0,2^k)$ →

→ k бит → Выходной бинарный поток .

Для наглядной визуализации авторами использовались карты бинарных последовательностей. Т.е. построение графической области пикселей в которой каждый пиксел соответствует определенному биту потока. Причем если бит имеет значение 1, то пиксел имеет белый цвет, если бит имеет значение 0 – то черный. На рисунке 2 показан результат шифрования периодического бинарного потока с помощью системы (5).



a) исходная b) зашифрованая

Рис.2. Карта битовой последовательнсти

Из рисунка 2b видно, что биты, распределены достаточно равномерно, что говорит, что полученная последовательность является случайной и следователь алгоритм работает достаточно эффективно.

Такой метод визуализации позволяет наглядно увидеть равномерность или неравномерность распределения битов, что дает возможность сделать выводы об случайности полученной последовательности.

Для доказательства эффективности алгоритма шифрования используют совокупность методов определения меры близости заданной псевдослучайной последовательности к случайной. В качестве такой меры обычно выступает наличие равномерного распределения, большого периода, равной частоты появления одинаковых подстрок и т. д.

Один из самых наглядных тестов – тест на равномерное распределение частот появления каждого символа. Пусть $x_0, x_1, \mathbf{K}, x_{255}$ – последовательность различных байтов размерности m=256. Далее считается частота вхождения каждого байта в тестируемый бинарный массив. На рисунке 3 показан результат частотного теста. Для этого был сформирован бинарный поток длиной 300000 и преобразован с помощью системы (3). График показывает частоту вхождения каждого байта в полученную последовательность.

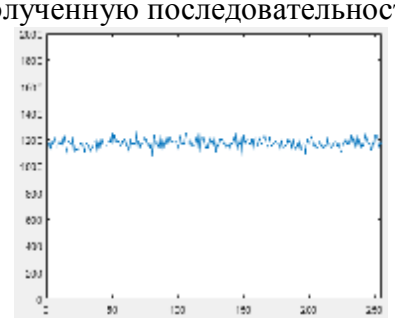


Рис.3. Результаты частотного теста

Частотный тест показывает, что каждый байт встречается в последовательности примерно равное число раз, что говорит о достаточно равномерном распределении символов в зашифрованной последовательности. Для оценки случайности последовательности чисел служат тесты, которые позволяют оценить, насколько исследуемый генератор случайных чисел «похож» или «не похож» на идеальную случайную последовательность. Такие тесты можно разделить на две группы.

Графические тесты результаты, которых отображаются в виде графиков, характеризующих свойства исследуемой

последовательности. Результаты таких тестов интерпретируются человеком, поэтому на их основе выводы могут быть неоднозначными.

Вторая группа – статистические тесты, выдают численную характеристику последовательности и позволяют однозначно сказать, пройден ли тест. Для оценки работы алгоритма шифрования на основе системы (5) авторами работы использовался пакет статистических тестов NIST. В его состав входят 15 статистических тестов, целью которых является определение меры случайности двоичных последовательностей, порождённых либо аппаратными, либо программными генераторами случайных чисел. Эти тесты основаны на различных статистических свойствах, присущих только случайным последовательностям. Большинство тестов было пройдено, или показали приемлемый результат, что говорит о высокой эффективности работы исследуемого алгоритма. Конкретный результаты и их анализ планируется опубликовать в отдельной статье.

Проблема корректной реализации операций в кольце целых чисел. При программировании вычислений на микроконтроллере используется модулярная алгебра. Т.е. все вычисления производятся в конечном кольце вычетов по некоторому натуральному модулю m, величина которого зависит от разрядности системы. Однако при реализации таких вычислений средствами стандартных математических операций возникает следующая проблема. Принцип работы функций модульной арифметики заключается в следующем: к операндам применяется соответствующая обычная (немодульная) функция, а затем результат делится с остатком на модуль. В этом случае размер промежуточных результатов может достигать 2MAX разрядов от входных данных, а в случае вычитания могут появляться отрицательные числа, что не допустимо. Такую ситуацию называют переполнением и потерей значащих разрядов соответственно. То есть, например, для реализации автомата с 8 битным входом и 8 битным входом необходим 16 битный контроллер. Там способом даже на современных 64 битных системах можно обрабатывать не более чем 32 битные данные. Механизм обработки таких ситуаций реализовать в десятичной системе без использования огромных процессорных ресурсов невозможно, что делает работу такого алгоритма очень неэффективной. В программном комплексе для реализации и тестирования алгоритма шифрования авторами реализован автомат, входом и выходом которого являются бинарные массивы, а все промежуточные результаты имеют тот же разряд что и все остальные данные. Алгоритм работает по следующей схеме.

Входной бинарный поток → k бит → $L(key, x^{in})$ → k бит → Выходной бинарный поток .

Такой подход позволяет не зависеть алгоритму от разрядности контролера, а только от размера буферной памяти.

Еще одной ***проблемой алгоритма защиты данных*** является проблема возникновения помех и искажений в зашифрованном сигнале. При передаче зашифрованного сигнала по каналам связи, особенно по радиоканалам, могут возникать помехи и искажения данных, что делает невозможным восстановление таких данных с помощью обратного автомата. В данном случае возникает две задачи. Определить, что в данных присутствуют искажения. Описать работу алгоритма при наличии искаженных данных. Данные задачи планируются быть исследованы авторами в будущих работах.

При шифровании бинарного потока последовательно считываются k бит, которые затем преобразуются с помощью прямого автомата. Если длина такого потока не кратна k, то перед последним шагом останется несколько бит меньше k. Возникает следующая проблема последних значений: как их обрабатывать? На-

пример, недостающие разряды можно заполнить нулями и полученные таким способом k бит передать в автомат. Однако, как этот может быть использовано для несанкционированного доступа к данным при криптоанализе системы?

Выводы

Любую управляемую динамическую систему, имеющую структуру вход-выход, можно использовать непосредственно для преобразования информации. Идея применения обратных систем управления со сложным поведением траекторий лежит в основе задачи синтеза новых эффективных алгоритмов защиты информации, в первую очередь, от несанкционированного доступа.

В настоящей работе предложен способ шифрования, использующий динамическую нелинейную систему трех уравнений (3). С программой шифрации - дешифрации был выполнен ряд экспериментов по преобразованию данных, в результате которых выявлен ряд проблем, относящихся к реализации алгоритма в микропроцессорных системах. Рассмотрен переход от непрерывного времени к дискретному для динамических систем вида (1). Записаны их дискретные аналоги, и приведен пример шифрования бинарных данных с помощью этих систем.

Список литературы

1. Обобщенная обратимость динамических систем в задачах шифрования / Ковалев А.М., Козловский В. А., Щербак В. Ф. – ПДМ, 2009, приложение № 1. – С. 20-21.

2. M. J. Sobhy and A. Shehata, Secure computer communication using haotic algorithms. Int. J. of Bifurcation and Chaos. vol. 10, no. 12, 2000, PP. 2831–2839.

3. Kirichenko V.V. Information Security of Communication Channel with UAV // Electronics and control systems. – 2015. – № 3. – С. 23-27.

4. Кириченко В.В., Лесина Е. В. Особенности информационной системы управления БПЛА // Наукові праці Донецького національного технічного університету. Серія : Інформатика, кібернетика та обчислювальна техніка. – 2016. – Вип. 1. – С. 111-116.

5. Кириченко В.В., Лесіна Є. В. Особливості дискретних алгоритмів перетворення інформації за допомогою обернених динамічних систем // Вісник черкаського університету. – Т. 389, №1. – С. 82-89.

Статью предоставлено в редакцию 29.09.2017