Математические структуры и моделирование
2015. № 4(36). С. 134–144

УДК 004.9

ИСПОЛЬЗОВАНИЕ ТУРБО-КОДЕКА ДЛЯ БЕЗОПАСНОЙ ПЕРЕДАЧИ ДАННЫХ

В.С. Виноградов
студент, e-mail: it.vinogradov@yandex.ru
В.В. Коробицын
доцент, к.ф.-м.н., e-mail: korobits@rambler.ru
М.Н. Московцев
преподаватель, e-mail: mnorthwind@gmail.com

Омский государственный университет им. Ф.М. Достоевского

Аннотация. В современных информационных системах, связанных с передачей информации по различным каналам связи, очень остро строит вопрос обеспечения информационной безопасности передаваемых данных. В работе предложен алгоритм защиты данных от несанкционированного доступа при передаче в каналах связи с помехами, основанный на алгоритмах класса турбо-кодов. Рассмотрены варианты использования подобного алгоритма, эффективность прикладной реализации, а также вспомогательная информационная инфраструктура для разработанного метода, например, алгоритм генерации ключей.

Ключевые слова: компьютерная безопасность, турбо-коды, турбо-декодер, генерация ключей, алгоритмы защиты данных.

Введение

Современные информационные технологии (например, сотовые сети стандартов 3G [1], 4G [2] или спутниковая радиосвязь [3]) прошли огромный эволюционный путь за последнее десятилетие, а их распространение подтолкнуло развитие систем и средств передачи данных. При этом коммуникационные технологии в определённых условиях должны работать в средах с низким соотношением сигнал-шум, что обеспечивает необходимость использования помехоустойчивого кодирования. Одним из наиболее востребованных на настоящий момент классов помехоустойчивых кодов является класс турбо-кодов. Также нельзя не отметить, что в современном мире все большую актуальность приобретает вопрос защиты информации от несанкционированного доступа или модификации в процессе передачи по сетям связи.

Цель статьи — изучение особенностей передачи данных в средах с низким соотношением сигнал-шум при использовании помехоустойчивого кодирования с помощью алгоритмов класса турбо-кодов, а также изучение аспектов защиты передаваемых таким образом данных.

1. Модель канала передачи данных

Источник данных на выходе обычно выдаёт поток бит информации. Далее следует этап подготовки сообщения для отправки по зашумлённому каналу связи, который включает в себя выполнение двух основных (интересующих нас в контексте данной статьи) операций: сжатие информации и применение помехоустойчивого кодирования.

Сжатие информации необходимо для уменьшения избыточности сообщения, что позволяет снизить затраты на передачу информации. Далее производится отправка данных по каналу связи. После осуществления передачи данных по каналу нужно провести обратную операцию, которая называется декодирование.

Общая схема модели канала передачи данных в средах с наличием шумов представлена на рис. 1.

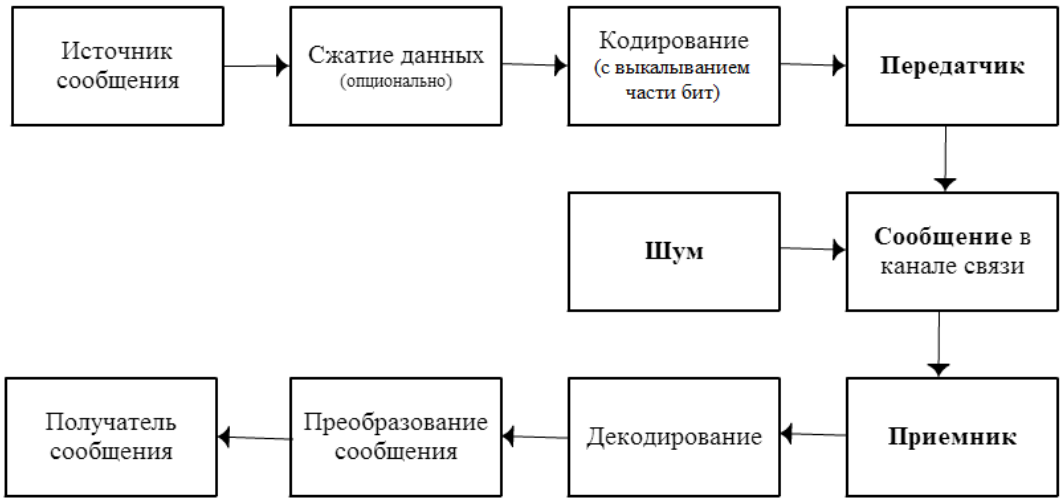

Рис. 1. Общая схема модели канала передачи данных

2. Введение в турбо-коды

Приведём базовые теоретические сведения об особенности работы и реализации турбо-кодов на практике, необходимые для понимания исследовательской части данной работы.

Турбо-код – это параллельный каскадный блочный систематический код со способностью исправления ошибок. Он является классом высокоэффективных помехоустойчивых кодов. Турбо-коды успешно применяются в спутниковой, цифровой связи и электротехнике, эффективно решая задачи достижения максимальной скорости передачи данных в зашумлённом канале в ограниченной полосе частот.

Основным преимуществом турбо-кодов является то, что они позволяют при разумных вычислительных затратах приблизиться к границе Шеннона [4], ко-

торая представляет собой теоретический предел помехоустойчивости. Также характерной особенностью является то, что сложность декодирования сообщения при использовании турбо-кодов не зависит от длины самого блока передаваемых данных, что позволяет снизить вероятность ошибки декодирования с помощью увеличения длины блока.

Главным же их недостатком является высокая сложность декодирования.

Программная реализация турбо-кодера включает в себя непосредственно турбо-кодер и турбо-декодер.

Параллельный турбо-кодер (рис. 2) имеет в своей структуре два компонентных кодера, перемежитель и мультиплексор.

Рис. 2. Схема турбо-кодера

Компонентные коды чаще всего являются рекурсивными систематическими свёрточными кодами (RSC). Важно заметить, что все входные биты передаются напрямую в выходной поток (далее информационные или систематические биты). Биты выходных потоков кодера называются паритетными.

Мультиплексор используется для слияния битов из трёх выходов в один выходной поток. Он размещает биты по определённому алгоритму, например, первый систематический бит, первый бит из первого потока паритетных битов, первый бит из второго потока паритетных битов и так далее.

В состав турбо-декодера входят демультиплексор, два компонентных декодера, соединённых через перемежитель и деперемежитель (рис. 3).

Декодер получает на вход отчёты, соответствующие принятому из канала связи демодулированному сигналу. Демультиплексор разделяет входные биты на «систематические биты», «паритетные биты №1» и «паритетные биты №2». На вход каждого компонентного декодера подаётся 3 потока бит: систематические, паритетные и поток от другого компонентного декодера.

Турбо-декодер работает итеративно. На вход первого декодера подаётся поток бит, принятый из канала связи, при этом значения априорной информации неизвестны. Выход первого декодера используется для вычисления априорной информации для второго декодера. Второй декодер принимает на вход информацию из канала связи, рассчитанную для него, а также априорную информацию от первого декодера. Далее начинается вторая итерация.

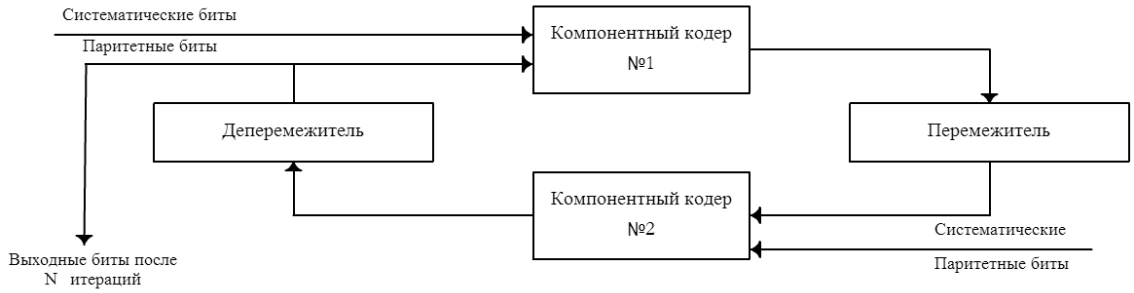

Рис. 3. Схема турбо-декодера

3. Использование турбо-декодера для защиты данных при передаче

Турбо-коды активно используются для передачи данных в средах с низким соотношением сигнал-шум и большим влиянием помех в каналах связи. Инкапсуляция несущего сигнала в шум сама по себе может служить некоторым средством защиты передаваемых данных, потому что она затрудняет отделение информационных битов от помех. Но это не даёт достаточного уровня защиты передаваемых данных (в случае перехвата сигнала при передаче).

Очевидным выглядит использование шифрования критически важных данных с помощью известных стойких криптографических алгоритмов перед процессом кодирования. Но нужно заметить, что шифрование больших объёмов данных (а тем более с последующим кодированием) является вычислительно ресурсоёмкой операцией. Поскольку мы не можем отказаться от кодирования, первоначальной идеей стало объединение этих двух процессов в один для сокращения вычислительных затрат.

Было предложено 2 пути использования турбо-декодера для защиты информации. Их использование целесообразно для разных типов сообщений и разных каналов передачи данных. Отправной точкой для определения этих вариантов стал график зависимости скорости кодирования при достоверном декодировании сообщения от соотношения сигнал-шум (рис. 4), из которого видно, что мы можем строить либо шумоподобную систему с низкой скоростью кодирования, либо систему с повышенной скоростью передачи данных при высоких соотношениях сигнал-шум.

4. Варианты построения защищённых систем

Первым вариантом использования турбо-декодера для защиты передаваемых данных стала организация шумоподобной системы. Особенность такой системы состоит в чрезвычайно низких соотношениях сигнал-шум (левая часть кривой на рис. 4), что требует применения сильных помехоустойчивых кодов. Следствием использования высокой избыточности является низкая скорость передачи данных (при, как правило, небольших размерах сообщения). При этом

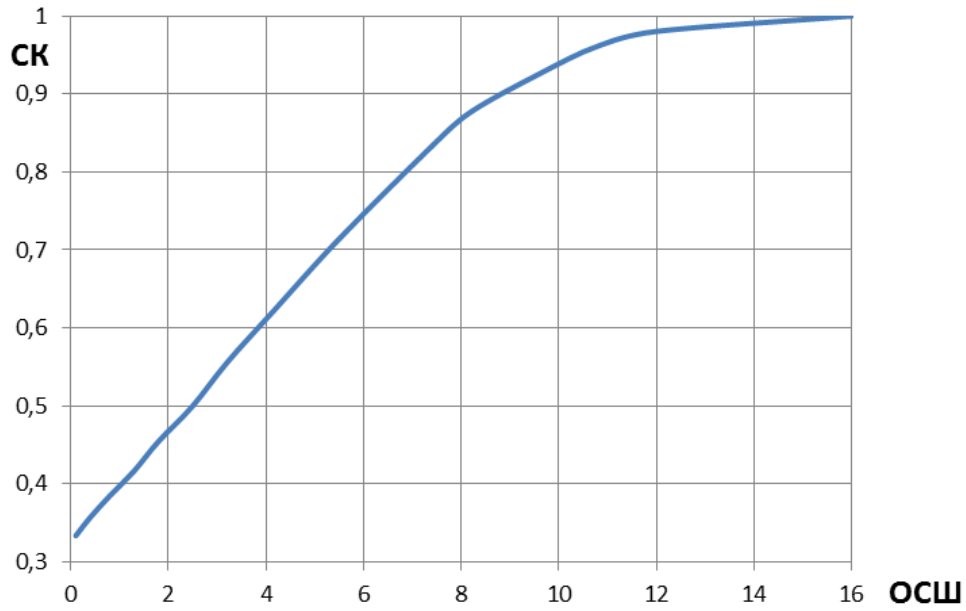

Рис. 4. Зависимость скорости кодирования от соотношения сигнал-шум при достоверном декодировании сообщения

помехи выполняют определённую маскирующую роль. Сообщение инкапсулировано в шум, что затрудняет выделение несущих битов из перехваченного потока. Построение таких систем имеет весьма неплохие перспективы и востребовано для реализации определённых задач на практике. Однако больший интерес представляло исследование систем с повышенной скоростью передачи данных.

В таких системах мы по-прежнему вынуждены использовать помехоустойчивое кодирование, поскольку влияние шумов на передаваемое сообщение остаётся значительным. Однако инкапсуляция информационных битов в шум уже не является достаточно надёжным методом защиты от раскрытия сообщения в случае перехвата.

На практике нужно обеспечивать более высокую скорость передачи данных, по сравнению с первым вариантом (из-за больших размеров сообщений и большей стоимости операции шифрования перед передачей). Для увеличения скорости применяется алгоритм выкалывания части паритетных битов (например, выкалывается каждый второй проверочный бит) из сообщения перед отправкой, что позволяет уменьшить количество передаваемых бит.

5. Постановка задачи

Требовалось обеспечить выполнение следующих задач:

1. Исключить передачу информационных бит по каналу связи в открытом виде;

2. Ввести параметры алгоритма, позволяющие на должном уровне обеспечить уникальность логического канала для конкретного сеанса связи;
3. Исследовать восстанавливающую возможность алгоритма в данных условиях.

Для решения первой задачи было предложено не передавать исходное сообщение вместе с проверочными битами (как это происходит при обычном кодировании информации). Была выработана идея восстанавливать оригинальное сообщение, используя только полученные из канала паритетные биты. Практическую реализацию этой задумки можно осуществить с помощью операции выкалывания. В существующей реализации турбо-декодера использовалось выкалывание каждого второго паритетного бита. Было принято решение выколоть все информационные биты и исследовать работу алгоритма в случае восстановления сообщения только по паритетным битам.

6. Выбор ключа для разрабатываемого алгоритма

Под ключом понимаются параметры алгоритма, которые будут обеспечивать уникальность работы алгоритма для конкретного соединения при передаче данных. В качестве ключа можно использовать параметры используемого кодера или перестановку перемежителя.

Напомним, что параметры используемого кодера определяют позиции выводов между сдвиговыми регистрами для каждого выходного потока и их изменение вызывает изменение выхода алгоритма кодирования.

Изменение перестановки также вызывает изменение выходного потока, поскольку выходной поток из перемежителя подаётся на компонентный кодер №2, который участвует в формировании выходного потока турбо-кодера путём мультиплексирования результатов работы обоих компонентных кодеров.

При выполнении операции турбо-кодирования параметры кодера, обычно, остаются статическими для большей эффективности кодирования и декодирования (они подбираются специальным образом для конкретных условий). Поэтому было решено исследовать возможность использования в качестве ключа алгоритма перестановки перемежителя.

7. Результаты тестирования

Для проверки корректности работы описанного алгоритма было проведено моделирование. Необходимо было оценить восстанавливающую способность полученного алгоритма. Для этого были построены кривые помехоустойчивости (рис. 5) и проведено их сравнение с аналогичными кривыми (рис. 6) для обычного декодирования (с выкалыванием только половины паритетных бит).

В эксперименте использовался кодер со следующими параметрами: скорость кодера $R = \frac{1}{2}$, количество состояний 16 (4 блока задержки), генератор $G_1 = 23$, $G_2 = 35$, блочный перемежитель с размером 65536 бит. Цифры 1-18 определяют число итераций декодирования. Тесты были проведены 10 раз с округлением результатов.

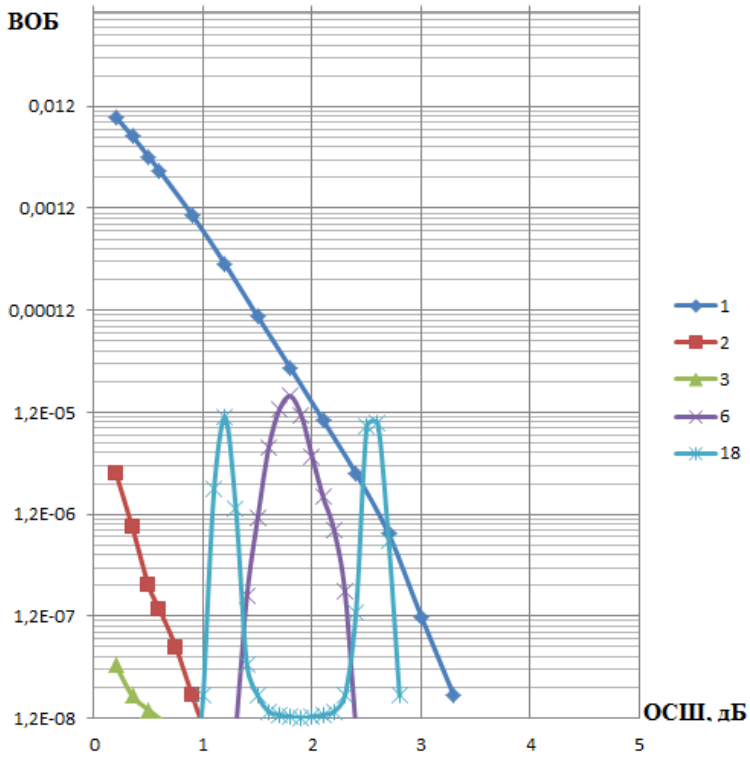

Рис. 5. График зависимости вероятности ошибки от соотношения сигнал-шум для системы с повышенной скоростью передачи данных

Как можно заметить из сравнения полученного графика с эталонным, при числе итераций от 1 до 3 декодер обеспечивает достаточное качество декодирования. Проблемы начинаются только при увеличении числа итераций. Предположительно, это происходит из-за того, что декодер не может только по паритетным битам восстановить информацию, и ошибки «размазываются» по другим битам в ходе следующих итераций декодирования. Решить эту проблему можно, отказавшись от использования более трёх итераций декодирования, либо отказавшись от использования декодера при определённых соотношениях сигнал-шум. Также потенциально более эффективным может оказаться использование усовершенствованного алгоритма декодирования с использованием хвостовых бит.

Второй проблемой оказался размер перестановки, которую предлагалось использовать в качестве ключа. Размер перестановки составляет 65536 бит. Использование ключа такого размера неприемлемо, поэтому было принято решение разработать алгоритм генерации перестановки из ключа меньшего размера.

8. Алгоритм генерации ключа

Размер перестановки определяется размером блока кодируемых данных. Его размер может достигать достаточно больших значений (порядка 65 кбайт в реальных условиях), что делает невозможным использование непосредствен-

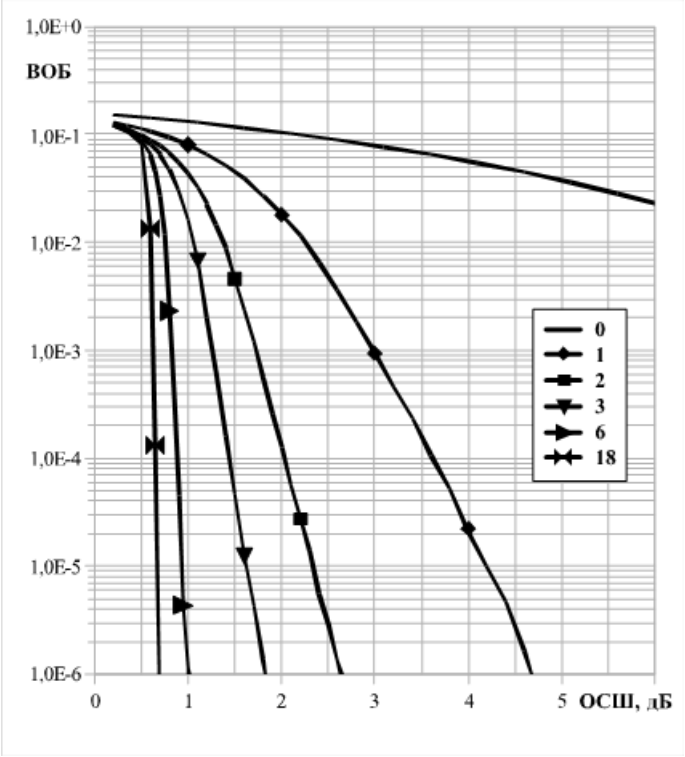

Рис. 6. Кривые помехоустойчивости для стандартного режима декодирования

но самой перестановки, например, из-за сложности передачи ключа, либо его хранения на устройстве. При этом размер ключа должен обеспечивать эффективную защиту от взлома методом прямого перебора.

Перестановка в памяти компьютера представляется как линейный уникальный массив без повторения элементов. Поэтому поставленная задача свелась к задаче генерации линейного уникального массива заданного размера.

Изначально в качестве тестового генератора перестановок был использован генератор псевдослучайных последовательностей, построенный по модели, предложенной канадским исследователем в области стохастического моделирования и оптимизации Пэрри Леквером (Pierre L'Ecuyer). Этот генератор использует метод перемешивания Байеса-Дарема (Bays-Durham Shuffling [5]) и имеет достаточную для наших задач длину периода. В качестве зерна генератора используется число типа long размерностью 4 байта (32 бита).

Было предложено в качестве ключа использовать зерно, а также внутренние параметры генератора. Очевидны следующие проблемы: параметры генератора подобраны специальным образом для максимальной эффективности алгоритма генерации псевдослучайных последовательностей. Поэтому использование случайных значений параметров не является эффективным. Длина ключа в 32 бита (ключ — зерно одного генератора) для современных алгоритмов не считается достаточной, поскольку позволяет осуществить взлом методом полного перебора за ограниченное время. Учитывая изложенные обстоятельства, от подобного подхода построения ключа было решено отказаться.

Был разработан алгоритм построения ключа, основанный на тех же принципах генерации перестановки перемежителя, которые использовались ранее. Суть алгоритма заключается в использовании нескольких генераторов псевдослучайных чисел. При этом ключ для алгоритма формируется с помощью конкатенации зёрен каждого генератора в бинарном виде.

Приведем описание алгоритма генерации перестановки с использованием сформированного ключа:

1. Берём полученный ключ (обозначим его длину за $keyL$) и разбиваем его на числа типа long, по 32 бита каждое. Количество таких чисел обозначим как $blockN$
2. Инициализируем массив perm, представляющий необходимую перестановку длины $permL$.
3. Инициализируем $blockN$ генераторов случайных чисел, в качестве зерен используем числа, полученные в пункте 1 алгоритма.
4. Вычисляем длину вспомогательных массивов $arrL$, $arrL = \frac{permL}{blockN}$. Данные массивы будут строиться с помощью генераторов случайных чисел (инициализированных в пункте 3). Каждый массив должен содержать уникальные числа от 1 до $arrL$.
5. Генерируем $(blockN - 1)$ вспомогательных массивов $arr[j](j = 1..arrL)$ длины $arrL$. Последний генератор, инициализированный в пункте 3, используется для расчёта $(permL - (blockN - 1) \cdot arrL)$ чисел, чтобы заполнить «конец» перестановки в случае, если длина перестановки не кратна $blockN$. При этом если число на выходе генератора не попадает в определённый в пункте 4 диапазон, оно просто отбрасывается.
6. Делаем пересчёт массивов, сгенерированных в пункте 5. Для j-го массива и его i-го элемента формула для пересчёта будет равна: $arr[j][i] = = arr[j][i] + (j - 1) \cdot (arrL)$.
7. Определяем первые $arrL \cdot (blockN - 1)$ элементов финальной перестановки по формуле $perm[j] = arr[i, k]$, где i = j mod $(blockN - 1)$, $k = \frac{j}{arrL}$. При этом если i = 0, оно принимается равным $(blockN - 1)$.
8. Формируем последний фрагмент необходимой перестановки, используя последний генератор псевдослучайных чисел. Для этого последовательно генерируем $(permL - (blockN - 1) \cdot arrL)$ чисел, лежащих в диапазоне от 1 до $(permL - (blockN - 1) \cdot arrL)$. Каждое число увеличиваем на $(blockN - 1) \cdot (arrL)$ и дописываем в конец массива perm последовательно. Выходы генератора, не попадающие в нужный диапазон, отбрасываются.
9. Полученный массив perm будет содержать готовую перестановку для перемежителя, используемого в алгоритме турбо-кодирования.

Наглядно алгоритм генерации перестановки по ключу представлен на рис. 7.

Как можно заметить, полученный ключ имеет следующие свойства:

1. Минимальная длина ключа — 64 бита. Данная особенность возникает из необходимости иметь как минимум 2 генератора псевдослучайных последовательностей с разным инициализирующим зерном. Это свойство не накладывает существенных ограничений на использование алгоритма.

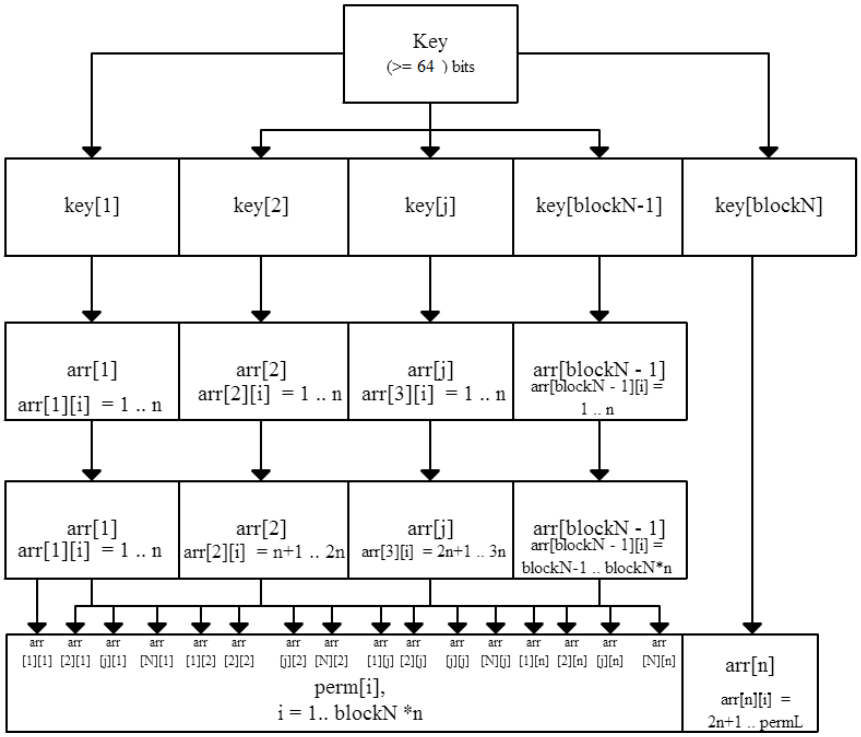

Рис. 7. Схема алгоритма генерации перестановки по ключу

2. Длина ключа должна быть кратна 32 битам.
3. Длина ключа сверху никак не ограничивается алгоритмом, но при этом необходимо отметить, что использование слишком больших ключей не является целесообразным.

При этом подобный алгоритм легко поддаётся распараллеливанию, что является очень удобным для организации системы управления ключами. Таким образом, с помощью алгоритмов параллельных вычислений можно формировать банки ключей для подобных систем связи.

Литература

1. An Investigation of Code Matched Interleaver for 3G Turbo Code Systems [Электронный ресурс]. URL: `http://www.researchgate.net/profile/Balamuralithara_Balakrishnan/publication/26560302_An_Investigation_of_Code_Matched_Interleaver_for_3G_Turbo_Code_Systems/links/00b4953070a93a2ce3000000.pdf` (дата обращения: 03.03.2015).
2. Особенности стандарта LTE [Электронный ресурс]. URL: `http://cmpo.vlsu.ru/edu/2013/A7.pdf` (дата обращения: 05.03.2015).
3. Спецификация DVB-RCS [Электронный ресурс]. URL: `https://www.dvb.org/resources/public/factsheets/DVB-RCS2_Factsheet.pdf` (дата обращения: 14.03.2015).

4. Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. М. : Техносфера, 2005. 320 с.
5. Метод перемешивания Байеса-Дарема [Электронный ресурс]. URL: `http://en.wikibooks.org/wiki/Statistics/Numerical_Methods/Random_Number_Generation#Bays-Durham_Shuffling_of_Uniform_Deviates` (дата обращения: 10.05.2015).

TURBO CODEC USING FOR THE SECURITY DATA TRANSMISSION

V.S. Vinogradov
Student, e-mail: it.vinogradov@yandex.ru
V.V. Korobitsin
Ph.D.(Phys.-Math.), Associate Professor, e-mail: korobits@rambler.ru
M.N. Moskovtsev
Instructor, e-mail: mnorthwind@gmail.com

Omsk State University n.a. F.M. Dostoevskiy

Abstract. The question of safe data transmission is very actual in the modern communication systems. In this article we offer the algorithm of safe data transmission which is based on turbocodes class algorithms. Options of such algorithm using, efficiency of current implementation and some information infrastructure for the developed method (for example, algorithm of key generation) are considered.

Keywords: computer security, turbo-codes, turbo-decoder, key generation, data protection algorithms.