УДК 381.3

Построение схемы динамического поиска структурированного блока данных в условиях неопределенности входной информации

Рассматривается схема динамического поиска структурированного блока данных в условиях неопределенности входной информации, которая при опросе сопрягаемых внешних устройств сравнивает эмпирические показатели функционирования системы с эталонным значением, заложенным в блоке памяти, и с помощью управляющих сигналов адаптирует работу системы управления под эталон.

А. Н. Моксяков, А. П. Мартынов, И. Г. Машин, Д. Б. Николаев, А. В. Седаков*

Внедрение информационных технологий позволяет строить системы управления и контроля, создавая разнородные контуры управления, включающие различные по своим функциональным и техническим характеристикам исполнительные устройства. В этом случае система управления должна гибко подстраиваться под формат и структуру данных, получаемых от исполнительных устройств. Зачастую это происходит загрузкой таблицы идентификации, соответствующей активному исполнительному устройству. Однако если невозможно определить источник входных данных, необходимо последовательно загрузить все таблицы идентификации для определения конкретного устройства. В случае низкопроизводительных систем управления этот процесс занимает недопустимо большой временной промежуток.

При рассмотрении вопросов управления в условиях неопределенности входной информации необходимо проанализировать основные методы управления неопределенными объектами, классификация которых приведена на рис. 1.

Робастные (грубые) системы – это системы управления, обеспечивающие приемлемое (в смысле некоторого критерия) качество при наличии параметрических, сигнальных, функциональных или структурных неопределенностей объекта управления. При этом, как правило, в ходе рабочего функционирования системы коэффициенты регулятора не подстраиваются, а малая чувствительность (т. е. грубость, или робастность) к различного рода вариациям математической модели объекта достигается за счет специальным образом выбранной структуры регулятора (алгоритма управления).

* 27 ЦНИИ МО РФ.

Таким образом, робастные системы относятся к классу ненастраивающихся систем управления, а их малая чувствительность к различного рода вариациям математической модели объекта обеспечивается на этапе синтеза алгоритма управления [1].

Адаптивные (самонастраивающиеся) системы – это системы управления, обеспечивающие компенсацию параметрических, сигнальных, функциональных или структурных неопределенностей объекта управления за счет автоматической подстройки регулятора в ходе рабочего функционирования системы. Другими словами, адаптивные системы восполняют нехватку априорной информации об объекте управления в ходе рабочего функционирования. В этом смысле они могут также называться самообучающимися системами [2].

Анализ методов управления применительно к решению задачи динамического поиска блока информации, соответствующего входным данных в условиях неопределенности их поступления, выявил как наиболее нам подходящий принцип построения идентификационных адаптивных систем (или систем с косвенной адаптацией), основанный на использовании процедуры идентификации объекта, т. е. на получении оценок его параметров или динамических характеристик. Полученные оценки используются далее для расчета местоположения блока информации. Таким образом, в своей структуре идентификационные адаптивные системы содержат (рис. 2) блок (алгоритм) идентификации, вырабатывающий оценки q неизвестных параметров объекта управления (адрес блока информации), блок расчета параметров регулятора k (формирователь идентификатора адреса) и собственно настраиваемый регулятор (алгоритм обработки входной информации с использованием адресуемого блока информации). Очевидно, что при стремлении оценок параметров объекта к истинным свойства замкнутой системы будут приближаться к желаемым.

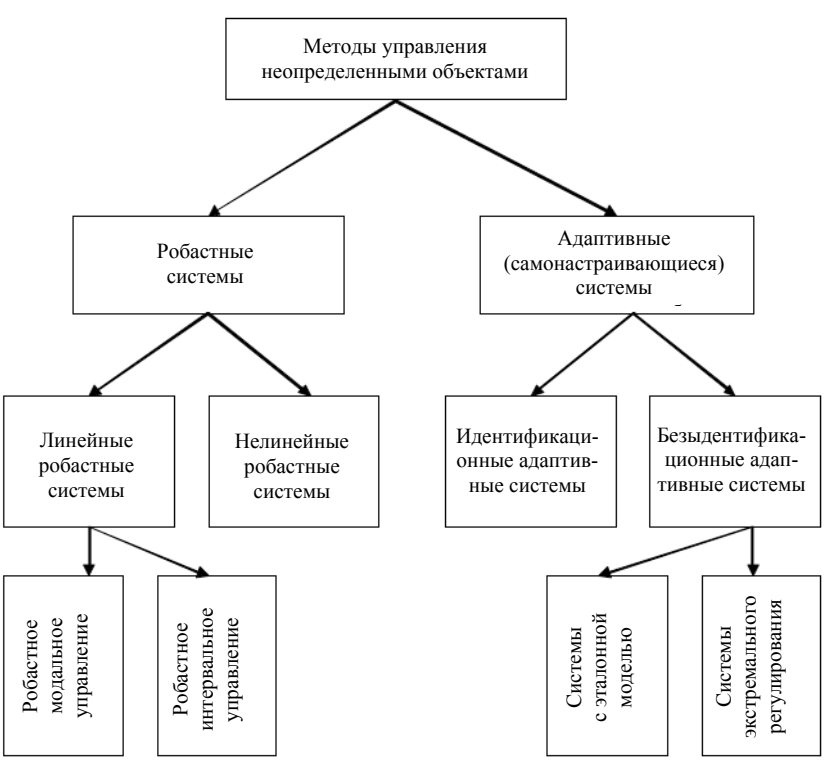


Рис. 1. Классификация методов управления неопределенными объектами

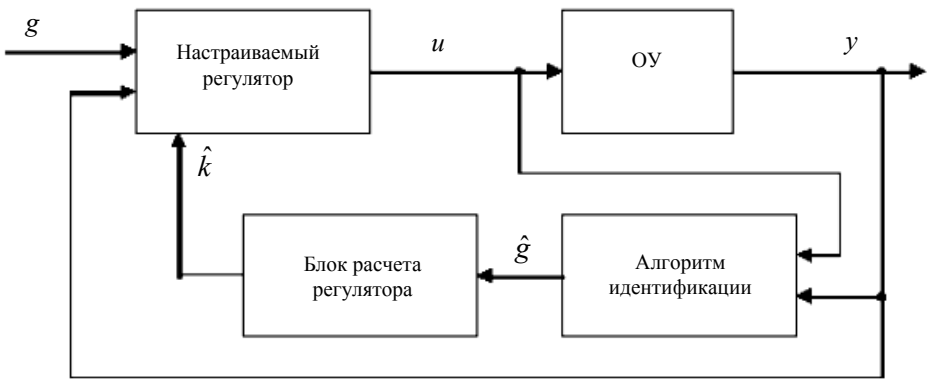


Рис. 2. Система идентификационного адаптивного управления

Несмотря на простоту основной идеи, системы с косвенной адаптацией обладают рядом существенных недостатков. Во-первых, описанная выше стратегия требует дополнительного времени на изучение объекта, что приводит к задержке при выработке правильного управления (обработки входных данных). Во-вторых, цели функционирования настраиваемого регулятора и блока оценки параметров оказываются, по существу, различными. Цель функционирования регулятора – обеспечение выполнения определенной последовательности действий, задаваемых входными данными g, в то время как цель блока идентификации – получение оценок параметров объекта управления (выявление истинного адреса блока информации, соответствующего входным данным). В этом смысле цепь настройки параметров оказывается разомкнутой по главной цели управления со всеми вытекающими отсюда негативными последствиями. В частности, большая ошибка в формировании результата обработки входных данных может никак не сказываться на скорости сходимости по параметрическим оценкам q и, в свою очередь, не ускорять процессы выбора соответствующего алгоритма обработки входных данных с использованием адресуемого блока информации [3].

Для устранения выявленных недостатков целесообразно применить метод хэширования, однозначно определяющий адрес блока информации по входным данным и настраивающий объект управления в соответствии с этим адресуемым блоком информации [4]. Хэширование – широко распространенный метод обеспечения быстрого доступа к информации, хранящейся в памяти. Записи распределяются между так называемыми сегментами, каждый из которых состоит из связного списка одного или нескольких блоков внешней памяти (рис. 3). Такая организация хэшированного файла подобна хэш-таблице с цепочками коллизий в памяти [5].

Имеется таблица сегментов, содержащая m указателей, по одному на каждый сегмент. Каждый указатель в таблице сегментов представляет собой физический адрес первого блока связного списка блоков для соответствующего сегмента. Сегменты пронумерованы от 0 до $m - 1$. Хэш-функция $h(k)$ отображает каждое значение ключа в одно из целых чисел от 1 до m. Если k – ключ, то $h(k)$ является номером сегмента, который содержит индекс записи с ключом k (если такая запись вообще существует). Блоки, составляющие каждый сегмент, образуют связный список. Таким образом, заголовок i-го блока содержит файловый указатель на $(i + 1)$-й блок. Последний блок сегмента содержит в своем заголовке *nul*-указатель. Каждый блок содержит несколько индексов с файловыми указателями на записи, хранящиеся в отдельном файле с данными.

Если размер таблицы сегментов невелик, ее можно хранить в основной памяти. В противном случае – в последовательных блоках в начале хэш-файла. Если нужно найти запись с ключом k, вычисляется $h(k)$ и определяется файловый указатель на элемент таблицы сегментов, со-

держащий указатель на первый блок списка сегмента. Затем последовательно считываются блоки из списка сегмента, пока не обнаружится блок, который содержит запись с ключом k. Если исчерпаны все блоки в связном списке для сегмента $h(k)$, то ключа k нет ни в одной из записей хэшированного файла [6].

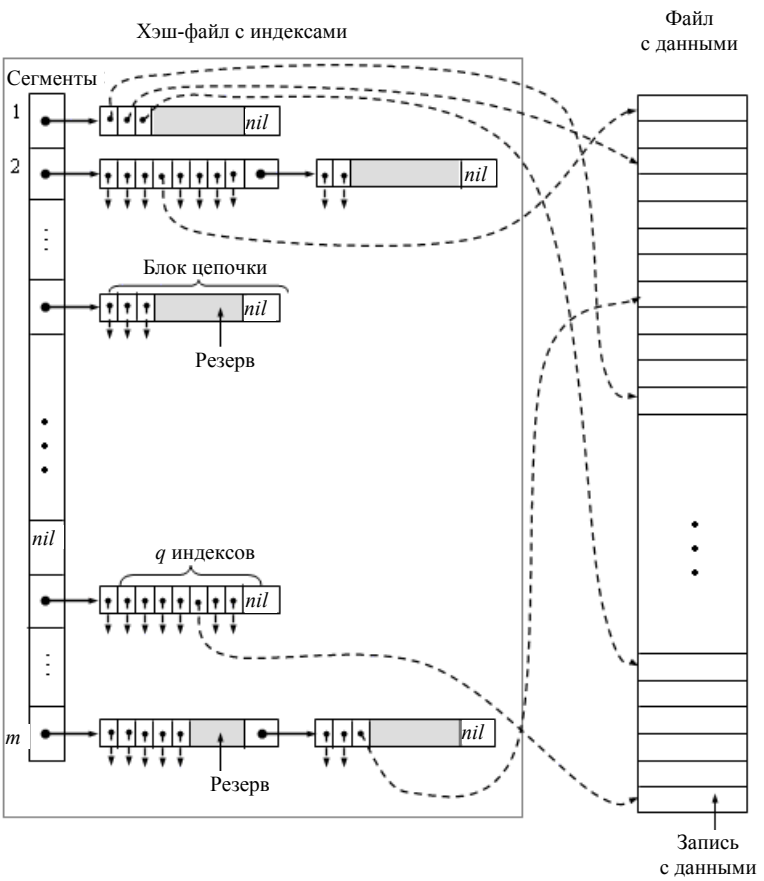


Рис. 3. Организация метода хэширования в файловой системе

Такая структура оказывается вполне эффективной при доступе к записям по ключу. Среднее количество обращений к блокам, требующееся для выполнения операций над таблицей, приблизительно равно среднему количеству блоков в сегменте, которое равно $n/(q \cdot m)$, если n – количество записей в файле с данными, q – количество индексов в блоке списка сегмента, m – количество сегментов. Операции поиска, вставки и удаления по ключу в хэшированном файле выполняются в среднем в m раз быстрее, чем в случае неорганизованного файла.

Чтобы вставить запись с ключом k, нужно сначала проверить, нет ли в файле записи с таким же значением ключа. Если она есть, то выдается сообщение об ошибке, поскольку предполагается, что ключ уникальным образом идентифицирует каждую запись. Если записи с ключом k нет, то вставляется новая запись в первый же блок цепочки для сегмента $h(k)$, в который эту запись удается вставить. Если запись не удается вставить ни в один из существующих блоков сегмента $h(k)$ ввиду отсутствия резерва в блоках, в хэш-файле создается новый блок, в который будет помещена эта запись. Этот новый блок затем добавляется в конец списка блоков сегмента $h(k)$.

Чтобы удалить запись с ключом k, нужно сначала осуществить ее поиск, а затем установить для нее бит удаления в файле с данными. Одновременно необходимо удалить индекс из соответствующего блока списка сегмента $h(k)$. Еще одной возможной стратегией удаления (которой нельзя пользоваться, если запись закрепленная) является замена удаленной записи на последнюю в цепочке блоков сегмента $h(k)$. Если такое изъятие приводит к опустошению последнего блока в сегменте $h(k)$, указатель на этот пустой блок можно сохранить в дополнительной структуре для повторного использования или вернуть файловой системе [7].

Хорошо продуманная организация файлов с хэшированным доступом требует лишь незначительного числа обращений к блокам при выполнении каждой операции. Если для функции хэширования выполняется условие $m = n/q$, тогда средняя длина списка сегмента равна единице. Если не учитывать обращения к блокам, которые требуются для просмотра таблицы сегментов, типичная операция поиска данных по ключу потребует лишь двух обращений к блоку сегмента и к записи в файле с данными, а операции вставки, удаления или изменения потребуют трех обращений. Если среднее количество записей в сегменте намного превосходит количество записей, которые могут уместиться в одном блоке, можно периодически реорганизовывать таблицу сегментов, удваивая количество сегментов и деля каждый сегмент на две части [8].

Предлагаемое решение для реализации динамического поиска относится к системе и способу, предназначенным для кодирования информации, идентифицирующей каждый блок информации, требуемой для обработки конкретного блока входных данных. Закодированный хэш-адрес определяется входной последовательностью и физическим адресом блока информации.

Способ получения хэш-адреса и его взаимосвязь с адресуемым блоком информации можно представить в следующем виде (рис. 4).

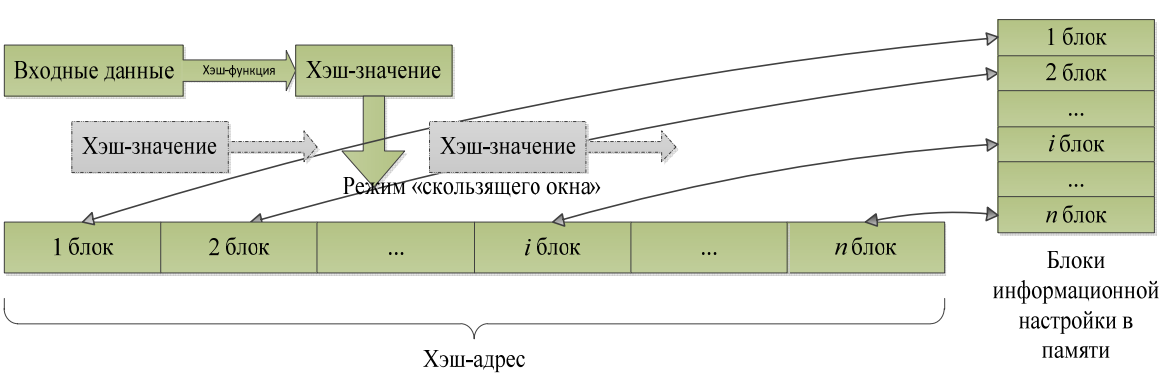


Рис. 4. Схема динамического поиска структурированного блока данных в условиях неопределенности входной информации

Для всего множества последовательностей входных данных формируются хэш-функции по одному из известных алгоритмов хэширования (SHA, MD-5, ГОСТ Р34.11-1997 и т. д.). Из полученных хэш-значений формируется хэш-адрес, который обеспечивает взаимно-однозначную связь адресуемого блока информации и конкретной последовательности входных данных. При этом не требуется сканирования всего объема информационных данных для идентификации блока с информацией, соответствующей входным данным. По входным данным формируется хэш-значение, которое в режиме «скользящего окна» сканирует хэш-адрес до получения совпадения. Совпавший блок однозначно адресует необходимый блок памяти с информацией для настройки обработки входных данных.

Рассмотрена схема динамического поиска структурированного блока данных в условиях неопределенности входной информации, которая при опросе внешних сопрягаемых устройств сравнивает эмпирические показатели с эталонным значением, заложенным в блоке памяти, и с помощью управляющих сигналов адаптирует работу системы управления под эталон.

Список литературы

1. Ахо А., Хопкрофт Д. Э., Ульман Д. Структуры данных и алгоритмы. М. – СПб – Киев: «Вильямс», 2000.

2. Каррано Ф. М., Причард Дж. Дж. Абстракция данных и решение задач на C++. Стены и зеркала. М. – СПб – Киев: «Вильямс», 2003.

3. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы. Анализ и построение. М.: «БИНОМ», 2000.

4. Грибунин В. Г., Мартынов А. П., Николаев Д. Б., Фомченко В. Н. Криптография и безопасность цифровых систем: Учебное пособие / Под ред. А. И. Астайкина. Саров: ФГУП «РФЯЦ-ВНИИЭФ», 2011.

5. Вирт Н. Алгоритмы и структуры данных. СПб.: «Невский диалект», 2001.

6. Кнут Д. Искусство программирования для ЭВМ. Т. 1. Основные алгоритмы. М.: «Вильямс», 2000.

7. Кнут Д. Искусство программирования для ЭВМ. Т. 3. Сортировка и поиск. М.: «Вильямс», 2000.

8. Левитин А. В. Алгоритмы: введение в разработку и анализ. М.: «Вильямс», 2006.

Construction of the Scheme of the Dynamic Search Structured Data Block in the Conditions of Uncertainty of Input Information

A. N. Moksjakov, A. P. Martynov, I. G. Mashin, D. B. Nikolaev, A. V. Sedakov

The report examines the scheme of the dynamic search structured data block in the conditions of uncertainty of the input information, which in a survey of the external periphery, compares empirical indicators with a reference value laid down in the block of memory and using control signals adapts work management system under the standard.

УДК 381.3

Разработка транслитерационного обратимого кодека для безопасной передачи информации по каналам связи общего пользования

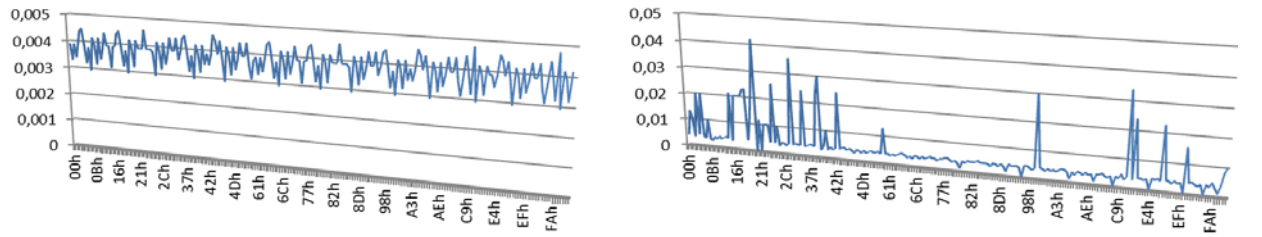
Рассмотрены вопросы построения транслитерационного обратимого кодека для безопасной передачи информации по каналам связи общего пользования. Предложен алгоритм транслитерации, реализуемый с использованием современных информационных технологий и не позволяющий осуществить раскрытие алгоритма и самой передаваемой информации.

Д. Б. Николаев, В. Г. Грибунин[1], С. Н. Колтаков[2], А. А. Скоробогатый[2], А. П. Мартынов

Глобализация информационного пространства, появление большого количества компаний, обеспечивающих различные виды связи, обусловили возможность создания резервных каналов передачи данных по линиям связи общего пользования. При этом необходимо обеспечивать безопасность и целостность передаваемой информации, что зачастую невозможно осуществить средствами, предоставляемыми компаниями-операторами.

В общем виде задача гарантированного обеспечения безопасности при передаче информации сводится к стойкому преобразованию, при этом в качестве преобразователя может выступать как классическая симметричная функция сокрытия, так и асимметричная функция. Результатом преобразования является псевдослучайная информационная структура с заданными характеристиками, базовой из которых является равномерность распределения информации внутри сообщения (кадра, фрейма). На рис. 1 представлены частотные характеристики исходного и преобразованного сообщений. Из рисунка видно, что преобразованные сообщения будут выделяться из всего информационного потока непреобразованных сообщений своей специфической структурой. Данный аспект снижает безопасность передачи преобразованных сообщений по линиям связи общего пользования из-за возможности применения простого вида фильтрации, основанного на частотных закономерностях алфавита сообщения. Кроме этого, современные средства обработки информации позволяют проводить анализ сообщения с использованием встроенных синтаксических и семантических словарей, что не позволяет заменять преобразованные сообщения со специфической структурой несмысловыми выражениями алфавита исходного текста, так как они могут быть отфильтрованы на этапе инкапсуляции в транспортные протоколы.

[1] МОУ «Институт инженерной физики», г. Серпухов.
[2] Генеральный штаб МО РФ.



а б

Рис. 1. Частотные характеристики исходного (а) и преобразованного (б) сообщений

Задачей проводимого исследования является создание транслитерационного обратимого кодека для безопасной передачи информации по каналам связи общего пользования. Общая схема применения «скрывающего» преобразования представлена на рис. 2.

Рис. 2. Схема применения «скрывающего» преобразования

На вход схемы подаются структурированные команды или управляющие последовательности заданного вида, структуру и смысловую нагрузку которых необходимо «скрыть» от «посторонних глаз». Эта входная информация обрабатывается преобразователем исходного текста [1], в качестве которого может использоваться реализация любого криптографического алгоритма (ГОСТ 28147-89, AES и т. д.) или алгоритма, формирующего псевдослучайные последовательности (RC-4, Dragon-128, HC-256 и др.). В результате преобразований данные будут представлены в виде последовательностей с характеристиками, близкими к равномерному распределению (псевдотекст). Как уже было сказано выше, данный тип информации совершенно не стоек к фильтрации, так как имеет отличную от других передаваемых сообщений структуру. Для устранения этого недостатка псевдотекст анализируется и преобразуется в блоке транслитерационного анализа (БТА) с использованием адаптивного полиалфавитного словаря (АПС). После этого полученное сообщение может быть передано в устройства формирования пакетов или другие транспортные системы для передачи, так как воздействие фильтрацией уже не приведет к желаемому результату [2]. На приемном конце полученное смысловое выражение претерпевает обратные преобразования при помощи блока транслитерационной декомпозиции (БТД), использующего АПС, аналогичный словарю для кодирования. Полученный псевдотекст восстанавливается соответствующими криптографическими методами в исходную структурированную команду или управляющую последовательность.

Рассмотрим работу транслитерационного обратимого кодека более подробно. Исходный псевдотекст представляет собой набор случайных символов и может быть представлен в виде последовательности шестнадцатеричных значений длиной l. Ее можно разбить на k блоков фикси-

рованной длины *d*, каждый из которых трансформируется при преобразовании в отдельную единицу (слово) смыслового выражения. В случае некратности псевдотекста принятым для кодирования размерам недостающий блок может быть дополнен нулевыми значениями (рис. 3).

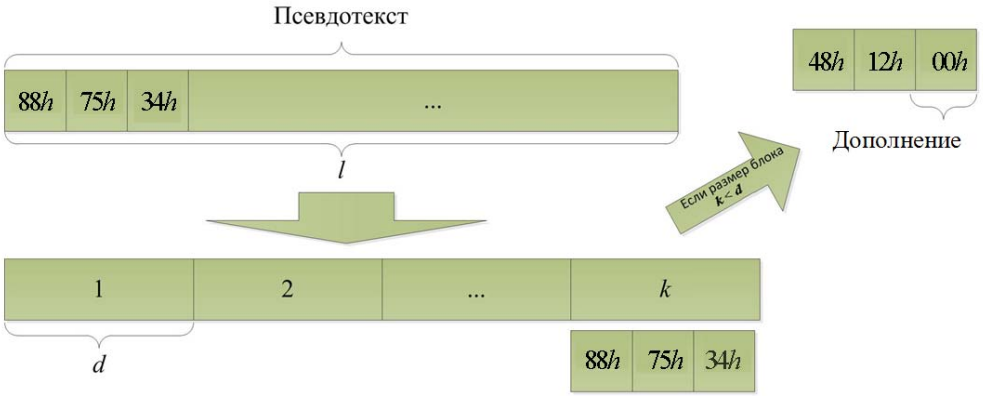


Рис. 3. Процедура подготовки сообщения (формирование блоков заданного размера) к транслитерационному преобразованию

Кодирование конкретного блока осуществляется с использованием АПС. Блок разбивается на две равные части, к каждой из которых добавляется поле контрольной информации (номер слова как функция от длины сообщения). Первая часть блока является номером строки, а вторая – номером столбца матрицы транслитерации, составляющей основу АПС. Ячейка с данными координатами содержит первое слово из смыслового выражения (рис. 4). Далее формируется множество разрешенных комбинаций, логически сочетающихся с первым словом сообщения.

Для оптимизации процедуры поиска приемлемых решений могут использоваться схемы динамического поиска структурированных информационных блоков. В нашем случае в структуре АПС с каждой ячейкой матрицы транслитерации связана лингвоформирующая комбинация (ЛФК), однозначно определяющая множество разрешенных комбинаций. После этого процесс кодирования повторяется для второго блока псевдотекста с той лишь разницей, что в случае непопадания координат блока в разрешенное множество происходит последовательное изменение сначала координаты столбца, а в случае неудачи – координаты строки до попадания в нужную ячейку. При этом для четных блоков происходит увеличение координаты на одну позицию, а для нечетных блоков – уменьшение (рис. 5).

По окончании кодирования второго блока формируется множество разрешенных комбинаций, логически сочетающихся с выражением, состоящим из первого и второго слов сообщения. И процесс кодирования третьего блока осуществляется по вышеописанному сценарию. В результате преобразования всех блоков сообщения формируется логическое лингвистически и синтаксически правильное сообщение, реализующее случайный псевдотекст. Мнемосхема с примером, иллюстрирующим работу транслитерационного обратимого кодека, показана на рис. 6.

Восстановление информации происходит в обратном порядке с использованием ЛФК, упрощающих поиск слова в матрице транслитерации. Корректирующий сдвиг однозначно вычисляется с помощью поля контрольной информации (ПКИ) по номеру блока.

Для расширения функциональности возможно использование внешних семантических корректоров с целью проверки согласования словосочетаний. Такие анализаторы выявляют подозрительные семантические конструкции в тексте и сообщают об этом, тем самым гарантируя отсут-

ствие в тексте определенных классов содержательных ошибок, выявление которых традиционными методами – весьма трудоемкий процесс. Перечень свойств, выявляемых корректором, включает правильность цепочек от задания до использования, а также инварианты, представляющие собой множество равенств термов, истинных в любом состоянии [3]. В этом случае в матрице транслитерации все слова могут располагаться в именительном падеже, что значительно увеличивает пространство возможных комбинаций и повышает безопасность разработанного способа транслитерационного кодирования.

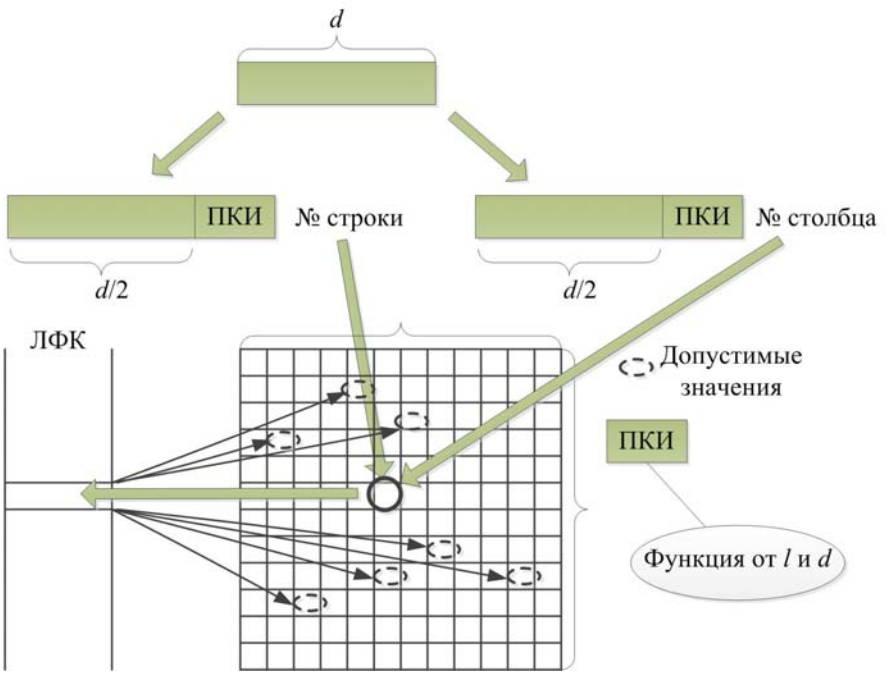


Рис. 4. Процесс формирования логически связанного смыслового выражения

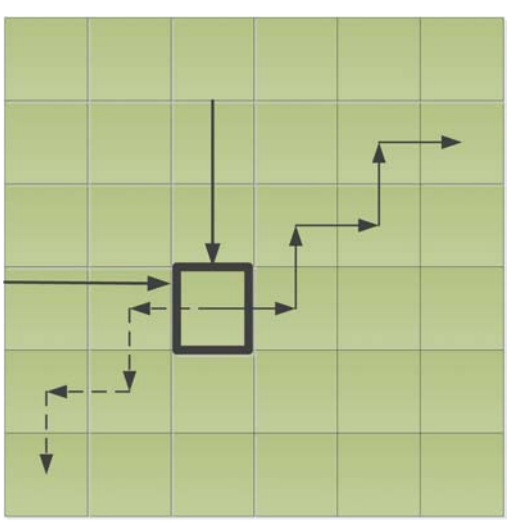


Рис. 5. Процедура поиска слова из разрешенного множества допустимых значений

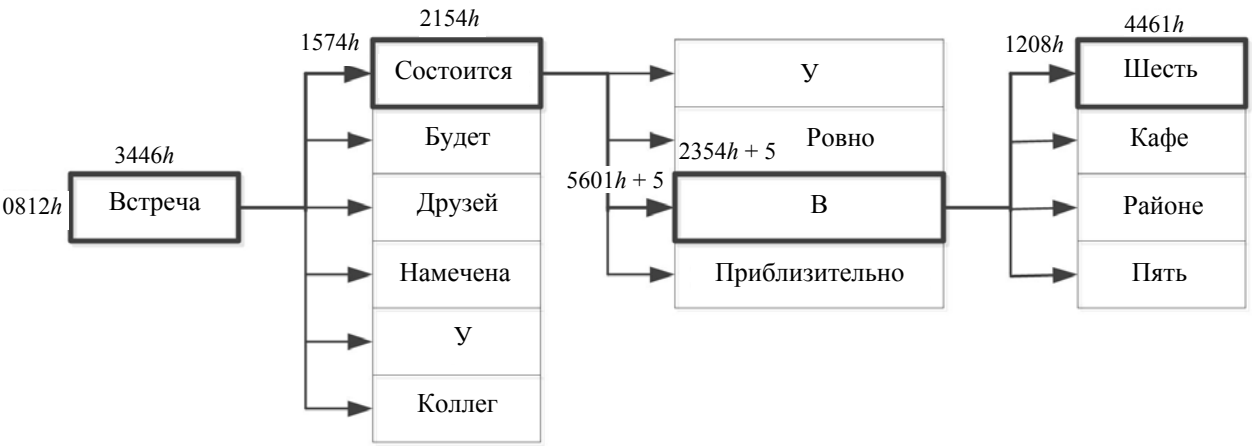


Рис. 6. Мнемосхема, иллюстрирующая работу транслитерационного обратимого кодека

Разработанный транслитерационный обратимый кодек является универсальным и может быть использован для решения ряда аналогичных задач, связанных с обеспечением безопасности информации и повышением удобства использования систем разграничения доступа. Например, условия применения практически всех систем паролирования для дистанционного доступа требуют запоминания достаточно длинного несмыслового пароля, представляющего собой набор разных сочетаний символов, в наилучшем случае, максимально использующем все символы ASCII-кода (0-255 значений (00*h*-*FFh*)). Предложенный кодек позволяет запоминать пароль в виде смысловой фразы, которая автоматически преобразуется в несмысловое случайное сообщение, символы которого выбраны из полностью используемого пространства ASCII-кодов, и циркулирует в системе уже в виде псевдотекста (рис. 7).

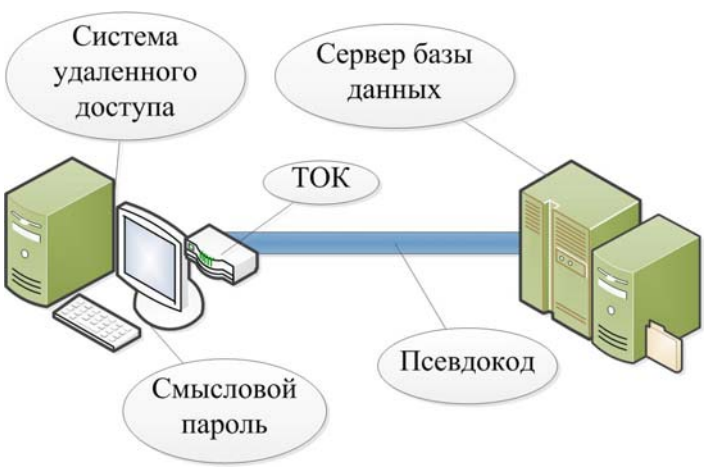


Рис. 7. Применение транслитерационного обратимого кодека в системах паролирования для дистанционного доступа к распределенным базам данных

Список литературы

1. Грибунин В. Г., Мартынов А. П., Николаев Д. Б., Фомченко В. Н. Криптография и безопасность цифровых систем: Учебное пособие / Под ред. А. И. Астайкина. Саров: ФГУП «РФЯЦ-ВНИИЭФ», 2011.

2. Кузьмин И. В., Кедрус В. А. Основы теории информации и кодирования: 2-е издание, перераб. и доп. К.: Вища шк., 1986.

3. Поттосин И. В. Российские исследования по языкам программирования и трансляции // МИР ПК – ДИСК, 2003, № 12. Студия программирования, стр. 1/16-16/16.

Working Out of the Transliteration Reversible Codec for Reliable Information Transfer Over Communication Channels of the General Using

D. B. Nikolaev, V. G. Gribunin, S. N. Koltakov, A. A. Skorobogatyj, A. P. Martynov

Questions of convertible codec construction for reliable transfer of the information on general purpose liaison channels are considered. The algorithm of a transliteration effectively enough sold with use of modern information technologies and not allowing to carry out disclosing of algorithm and the most transmitted information is offered.