ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2008 **№1(1)**

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Адрес редакции: 634050, г. Томск, пр. Ленина, 36
E-mail: vestnik_pdm@mail.tsu.ru

В журнале публикуются результаты фундаментальных и прикладных научных исследований отечественных и зарубежных ученых, включая студентов и аспирантов, в области дискретной математики и ее приложений в криптографии, компьютерной безопасности, кибернетике, информатике, программировании и теории надежности.

Периодичность выхода журнала: 4 номера в год.

ООО «Издательство научно-технической литературы»
634050, Томск, пл. Ново-Соборная, 1, тел. (3822) 533-335

Редактор *Н.И. Шидловская*
Верстка *Д.В. Фортес*

К-ОКП ОК-005-93, код продукции 952000

Изд. лиц. ИД № 04000 от 12.02.2001. Подписано к печати 18.06.2008.
Формат 70 × 100 $^{1}/_{16}$. Бумага офсетная. Печать офсетная. Гарнитура «Таймс».
Усл. п. л. 9,35. Уч.-изд. л. 10,48. Тираж 300 экз. Заказ № 7.

Отпечатано в типографии «М-Принт», г. Томск, ул. Пролетарская, 38/1

СОДЕРЖАНИЕ[1]

О ЖУРНАЛЕ....5

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

Горшков С.П. О сложности нахождения приведенных представлений слабо положительных и слабо отрицательных булевых функций 7
Иванов А.В. Мономиальные приближения платовидных функций....10
Кутьин А.М. Коды, композиции и решетки....15
Кыров В.А. Трехбазисные квазигруппы с обобщенным тождеством Уорда 21
Пудовкина М.А. Линейные структуры групп подстановок над конечным модулем....25
Шоломов Л.А. Энтропия недоопределенных последовательностей при ограничениях на доопределения....29

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

Агибалов Г.П. Элементы теории дифференциального криптоанализа итеративных блочных шифров с аддитивным раундовым ключом 34
Росошек С.К., Боровков А.А., Евсютин О.О. Криптосистемы клеточных автоматов....43
Токарева Н.Н. Квадратичные аппроксимации специального вида для четырехразрядных подстановок в S-блоках....50
Черемушкин А.В. Комбинаторно-геометрические подходы к построению схем предварительного распределения ключей (обзор)....55

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

Девянин П.Н. Базовая ролевая ДП-модель 64
Колегов Д.Н. Применение ДП-моделей для анализа защищенности сетей 71
Кучеров М.М., Кирко И.Н., Муллер А.А. Современные модели и механизмы защиты информации 88
Стефанцов Д.А. Реализация политик безопасности в компьютерных системах с помощью аспектно-ориентированного программирования....94

ПРИКЛАДНАЯ ТЕОРИЯ ГРАФОВ

Абросимов М.Б., Долгов А.А. Семейства точных расширений турниров 101
Евдокимов А.А., Левин А.А. Инструментарий графического исследования символьных последовательностей....108
Науменко И.А., Скобелев В.Г. Преобразование решёток в 1-отказоустойчивые графы....111
Салий В.Н. Минимальные примитивные расширения ориентированных графов....116

ПРИКЛАДНАЯ ТЕОРИЯ АВТОМАТОВ

Бушков В.Г. К описанию прогрессивных решений параллельного автоматного уравнения 120
Скобелев В.В. Характеристика неподвижных точек линейных автоматов над конечным кольцом....126
Сухинин В.А., Скобелев В.Г. Идентификация автомата в классе автоматов Спротта....131

СВЕДЕНИЯ ОБ АВТОРАХ....136

АННОТАЦИИ СТАТЕЙ НА АНГЛИЙСКОМ ЯЗЫКЕ 138

[1] Статьи этого номера представлены оргкомитетом VII Сибирской научной школы-семинара с международным участием «Компьютерная безопасность и криптография» – SIBECRYPT'08. Школа-семинар проведена совместно Томским госуниверситетом и Сибирским государственным аэрокосмическим университетом в сотрудничестве с Институтом криптографии, связи и информатики Академии ФСБ с 9 по 12 сентября 2008 года в г. Красноярске при финансовой поддержке РФФИ (грант № 08-07-06024-г).

CONTENTS

ABOUT THE JOURNAL 5

THEORETICAL BACKGROUNDS OF APPLIED DISCRETE MATHEMATICS

Gorshkov S.P. On the complexity of weakly positive and weakly negative Boolean functions reducing 7
Ivanov A.V. Approximation of plateaued Boolean functions by monomial ones 10
Kut'in A.M. Codes, compositions and lattices 15
Kirov V.A. Three-basal quasigroup with generalized word's identity 21
Pudovkina M.A. Linear structures of permutation groups over finite modules 25
Sholomov L.A. Entropy of underdetermined sequences under constraints to specifications 29

MATHEMATICAL METHODS IN CRYPTOGRAPHY

Agibalov G.P. Some theoretical aspects of differential cryptanalysis of the iterated block ciphers with additive round key 34
Rososhek S.K., Borovkov A.A., Evsutin O.O. Cellular automaton cryptosystems 43
Tokareva N.N. Quadratic approximations of the special type for the 4-bit permutations in S-boxes 50
Cheremushkin A.V. Combinatorial – geometric technique to design a key distribution patterns (an overview) 55

MATHEMATICAL BACKGROUNDS OF COMPUTER SECURITY

Devyanin P.N. Base role DP-model 64
Kolegov D.N. DP-model application for network security analysis 71
Kutcherov M.M., Kirko I.N., Muller A.A. Modern models and mechanisms in information security 88
Stephantsov D.A. Implementation of security policies in computer systems by means of aspect-oriented programming 94

APPLIED GRAPH THEORY

Abrosimov M.B., Dolgov A.A. Families of tournament's extensions 101
Evdokimov A.A., Levin A.A. A toolkit for graphical analysis of words chains 108
Naumenko I.A., Skobelev V.G. Transformation of grids to 1-fault-tolerant graphs 111
Salii V.N. Minimal primitive extensions of oriented graphs 116

APPLIED AUTOMATA THEORY

Bushkov V.G. Describing progressive solutions to a parallel FSM equation 120
Skobelev V.V. Characteristics of fixed points for linear automata over a finite ring 126
Sukhinin V.A., Skobelev V.G. Automaton identification in the class of Sprott's automata 131

BRIEF INFORMATION ABOUT THE AUTHORS 136

PAPER ABSTRACTS 138

О ЖУРНАЛЕ

Исследования по дискретной математике в Томском государственном университете (ТГУ) начались в конце 1950-х годов. Они были начаты тогдашним аспирантом радиофизического факультета (РФФ) Аркадием Дмитриевичем Закревским. Его первая научная статья в данном направлении вышла в 1959 г. в Докладах Академии наук СССР под названием «Метод синтеза функционально устойчивых автоматов». Тот факт, что эти исследования зародились на РФФ, а не на мехмате, наложил определенный отпечаток на их характер. Изначально они велись как прикладные, т.е. вызывались не потребностями самой математики, но ее применениями – главным образом, к синтезу дискретных автоматов (тогда – релейных схем управления, позднее – дискретных управляющих систем). Прикладной характер исследований выразился и в том, что их результатами должны были стать алгоритмы решения задач дискретной математики, аттестованные в эксперименте на электронной вычислительной машине (ЭВМ). А.Д. Закревский одним из первых (если не самым первым) в мировой науке предложил и осуществил на практике технологию экспериментального исследования алгоритмов на ЭВМ, ставшую отличительной особенностью и обязательным элементом деятельности его научной школы. Фактически уже тогда речь шла о создании нового направления в математике – компьютерной дискретной математики. И А.Д. Закревский заложил первые «кирпичи» в фундамент этой науки, предложив системы подмножеств конечных множеств представлять в ЭВМ при помощи булевых и троичных матриц и разработав иерархическую систему математических операций над такими матрицами и эффективные алгоритмы их выполнения. Система охватывает широкий спектр операций – от простейших (нахождение минимального столбца и максимальной строки) до предельно сложных (кратчайшее покрытие булевой матрицы или минимальное разбиение множества ее столбцов на совместимые подмножества, например). Фундаментальное и прикладное значение этой системы состоит в том, что через операции в ней легко выражаются алгоритмы решения многих задач как в самой дискретной математике, так и в ее приложениях – и не только к синтезу дискретных автоматов. К числу таких задач относятся: минимизация и функциональная разделимость булевых функций, минимизация конечных автоматов и кодирование их состояний, логический синтез схем и тестов для них, раскраска графов, градостроительные задачи размещения, трассировки и многие другие. Полезность этой разработки вскоре была доказана на практике, когда на ЛЯПАСе – «русском языке программирования», разработанном под руководством А.Д. Закревского, – в ТГУ был создан ряд систем автоматического синтеза и диагностики дискретных автоматов, основу математического обеспечения в которых составила именно данная система операций. В 70-е и 80-е годы XX века эти автоматические системы широко применялись на предприятиях МЭП и МРП СССР.

За полвека своего существования школа прикладной дискретной математики ТГУ освоила многие научные направления в этой области и достигла в них крупных результатов. Кроме уже упомянутых языка программирования и систем автоматического синтеза и диагностики дискретных автоматов, это и технология решения комбинаторно-логических задач методом сокращенного обхода дерева поиска, с помощью которой разработаны самые эффективные (на момент создания) алгоритмы для целого ряда таких задач, и теория конечных автоматов, где алгебраическими и комбинаторными методами решены многие задачи существования и характеризации нетривиальных декомпозиций, и теория дискретных автоматов на полурешетках, позволившая впервые поставить и решить задачи анализа и синтеза цифровых интегральных схем с заданным динамическим поведением и выяснить условия полноты, выразимости и схемной реализуемости дискретных функций на полурешетках; это и разработка и экспериментальное исследование теоретико-числовых алгоритмов в криптографии, алгоритмов анализа и синтеза булевых функций с криптографическими свойствами, алгоритмов решения систем уравнений над конечным полем и параллельных комбинаторных алгоритмов; это и синтез и криптоанализ поточных шифров, и решение автоматных уравнений, и синтез самопроверяемых и контролепригодных схем, и многое другое. В настоящее время исследования по прикладной дискретной математике в ТГУ направлены на создание, развитие и применение математических основ информатики и программирования, криптографии и компьютерной безопасности, логического проектирования и надежности, теории автоматов и вычислительных методов.

Результаты научных исследований по направлениям прикладной дискретной математики в стране и за рубежом регулярно обсуждаются на многочисленных научных форумах, в том числе на Сибирской научной школе-семинаре с международным участием «Компьютерная безопасность и криптография» – SIBECRYPT, проводимой ТГУ ежегодно, начиная с 2002 г. В работе школы-семинара принимают участие до сотни крупных ученых и специалистов научных учреждений, аспирантов и студентов вузов из разных городов России, Украины и Белоруссии, включая Москву, Санкт-Петербург, Донецк, Минск, Екатеринбург, Саратов, Омск, Новосибирск, Томск, Красноярск, Иркутск и др.

В настоящее время в России есть только два научных журнала по дискретной математике – «Дискретная математика» и «Дискретный анализ и исследование операций». Оба ориентированы на публикацию результатов, главным образом, фундаментальных научных исследований, и их объем ни в коей степени не соответствует размаху этой науки в стране. Как следствие, результаты многочисленных прикладных исследований в области дискретной математики, в том числе мирового уровня, часто не находят отражения в серьез-

ных научных изданиях. Это всецело относится и к тем интересным научным результатам, которые нередко представляются на школе-семинаре SIBECRYPT.

Естественно, что постоянный рост научной продукции «дискретных математиков-прикладников» вызывает значительное увеличение потребности в публикации их научных работ. Создание печатного органа в формате журнала «Прикладная дискретная математика» отвечает этой потребности. Все сказанное выше о развитии прикладной дискретной математики в ТГУ служит убедительным обоснованием правомерности размещения этого органа в Томском государственном университете.

Тематика и оргкомитет школы-семинара SIBECRYPT процентов на 90 совпадают соответственно с тематикой и редакционной коллегией журнала «Прикладная дискретная математика». Это дает основание оргкомитету школы-семинара доклады, получившие положительную оценку в обсуждениях на школе-семинаре, рекомендовать к возможному опубликованию в виде статей в этом журнале, а редакционной коллегии журнала – статьи, представленные по рекомендации оргкомитета школы-семинара, рассматривать в первую очередь.

Тематика публикаций в журнале охватывает 13 разделов прикладной дискретной математики со следующим их содержанием:

1) *математические основы информатики и программирования* – формальные языки и грамматики, алгоритмические системы, языки программирования, структуры и алгоритмы обработки данных, теория вычислительной сложности;

2) *вычислительные методы в дискретной математике* – теоретико-числовые методы в криптографии, вычислительные методы в теории чисел и общей алгебре, комбинаторные алгоритмы, параллельные вычисления, методы дискретной оптимизации, дискретно-событийное и клеточно-автоматное моделирование;

3) *математические основы криптографии* – алгебраические структуры, дискретные функции, комбинаторный анализ, теория чисел, математическая логика, теория информации, системы уравнений над конечными полями и кольцами;

4) *математические методы криптографии* – синтез криптосистем, методы криптоанализа, генераторы псевдослучайных последовательностей, оценка стойкости криптосистем, криптографические протоколы, математические методы квантовой криптографии;

5) *математические методы стеганографии* – синтез стеганосистем, методы стеганоанализа, оценка стойкости стеганосистем;

6) *математические основы компьютерной безопасности* – математические модели безопасности компьютерных систем (КС), математические методы анализа безопасности КС, математические методы синтеза защищенных КС;

7) *прикладная теория кодирования* – коды для сжатия данных и защиты информации, коды для обнаружения и исправления ошибок, построение оптимальных кодов, анализ свойств кодов;

8) *прикладная теория автоматов* – автоматные модели протоколов в компьютерных сетях, автоматные модели криптосистем, автоматные модели управляющих систем, автоматы без потери информации, эксперименты с автоматами, декомпозиция автоматов, автоматные уравнения, клеточные автоматы;

9) *логическое проектирование дискретных автоматов* (ДА) – математические модели ДА, математические методы анализа ДА, математические методы синтеза ДА, математические методы оптимизации ДА, оценка сложности ДА;

10) *математические основы надежности вычислительных и управляющих систем* (ВиУС) – математические модели функциональной устойчивости ВиУС (к отказам, неисправностям, сбоям, состязаниям, исследованию), математические методы анализа функциональной устойчивости ВиУС, математические методы синтеза функционально устойчивых ВиУС, математические методы верификации логических схем и программ, математические методы синтеза самопроверяемых и контролепригодных схем;

11) *математические основы интеллектуальных систем* – базы данных, базы знаний, логический вывод, экспертные системы;

12) *прикладная теория графов* – графовые модели в информатике и программировании, в компьютерной безопасности, вычислительных и управляющих системах, в интеллектуальных системах;

13) *исторические очерки по дискретной математике и ее приложениям* – в криптографии, компьютерной безопасности, кибернетике, информатике, программировании и теории надежности.

В журнале публикуются как оригинальные результаты, так и работы обзорного характера, написанные по заказу редакционной коллегии. Возможны тематические выпуски по любому из перечисленных разделов. Средний объем номера – 128 с. формата А4. Средний объем статьи с оригинальными результатами – 5 с., кратких сообщений – 2 с.

Председатель редакционной коллегии журнала
заведующий кафедрой защиты информации и криптографии ТГУ
профессор Г.П. Агибалов

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

DOI 10.17223/20710410/1/1

УДК 519.7

О СЛОЖНОСТИ НАХОЖДЕНИЯ ПРИВЕДЕННЫХ ПРЕДСТАВЛЕНИЙ СЛАБО ПОЛОЖИТЕЛЬНЫХ И СЛАБО ОТРИЦАТЕЛЬНЫХ БУЛЕВЫХ ФУНКЦИЙ

С.П. Горшков

Институт криптографии, связи и информатики Академии ФСБ России, г. Москва

E-mail: spg54@bk.ru

В работе оценивается сложность задачи построения приведенных представлений слабо положительных и слабо отрицательных булевых функций, записанных в совершенной конъюнктивной нормальной форме или многочленом Жегалкина.

Ключевые слова: *вычислительная сложность*, *слабо положительная* (*слабо отрицательная*) *булева функция.*

Булева функция $f(x_1, ..., x_k)$ называется:

1) *слабо положительной*, если $f \equiv 1$ или существует представление f в виде следующей КНФ:

$$f \equiv \bigwedge_{i=1}^{t} (x_{s_{i1}}^{\alpha_i} \vee x_{s_{i2}} \vee ... \vee x_{s_{ik_i}}), \quad (1)$$

где $\alpha_i \in \{0, 1\}$, $i = 1, ..., t$;

2) *слабо отрицательной*, если $f \equiv 1$ или существует представление f в виде следующей КНФ:

$$f \equiv \bigwedge_{i=1}^{t} (x_{s_{i1}}^{\alpha_i} \vee \overline{x}_{s_{i2}} \vee ... \vee \overline{x}_{s_{ik_i}}), \quad (2)$$

где $\alpha_i \in \{0, 1\}$, $i = 1, ..., t$.

Множество всех функций классов 1 и 2 обозначим соответственно *WP*, *WN*. Формулы (1), (2) соответственно для функций классов *WP*, *WN* будем называть *приведенными представлениями*.

В работах [1, 2] показано, что слабо положительные и слабо отрицательные функции (наряду с мультиаффинными и биюнктивными функциями) порождают полиномиально решаемые классы систем булевых уравнений без ограничений на выбор неизвестных. В англоязычной литературе слабо отрицательные функции часто называются хорновскими функциями, а слабо положительные булевы функции – антихорновскими.

Интерес к задаче оценки сложности нахождения приведенных представлений слабо положительных и слабо отрицательных функций по другим заданиям функций обусловлен следующими моментами.

1. В общем случае задача перевода функций из одного вида в другой (например, построение по многочлену Жегалкина функции совершенной дизъюнктивной нормальной формы этой функции) имеет экспоненциальную сложность.

2. Если все функции левой части системы булевых уравнений

$$\{ f_i(x_1, ...x_n) = 1, i = 1, ..., m, \quad (3)$$

слабо положительные (слабо отрицательные) функции и записаны в приведенном виде, то существует алгоритм линейной сложности (сложности $O(\text{len}(S))$, где $\text{len}(S)$ – длина записи системы (3)) распознавания совместности (3) [3].

3. Если две слабо положительные (слабо отрицательные) функции f_1, f_2 заданы в приведенном виде, то с полиномиальной сложностью можно определить $f_1 \stackrel{?}{\equiv} f_2$.

Напомним некоторые определения. Функция $h(x_1, ..., x_k)$ называется *имплицентой* функции $f(x_1, ..., x_k)$, если

$$f(x_1, ..., x_k) \wedge h(x_1, ..., x_k) \equiv f(x_1, ..., x_k).$$

Имплиценты вида $x_{s_1}^{\alpha_1} \vee x_{s_2}^{\alpha_2} \vee ... \vee x_{s_r}^{\alpha_r}$, где $s_i \neq s_j$ при $i \neq j$, называются *элементарными имплицентами*. Элементарная имплицента функции f называется *простой*, если никакая ее собственная часть не является имплицентой функции f.

Любая функция $f(x_1,...,x_k)$, не равная тождественно 1, представляется конъюнкцией всех своих простых имплицент. Конъюнктивная нормальная форма функции *f*, в которую входят все простые имплиценты функции *f* и только они, называется *сокращенной* КНФ функции *f*. Любая функция $f(x_1,...,x_k)$ имеет единственную, с точностью до перестановки сомножителей, сокращенную КНФ. Сокращенная КНФ слабо положительной (слабо отрицательной) булевой функции является одной из ее приведенных представлений.

Утверждение 1. Задача построения по совершенной КНФ (СКНФ) слабо положительной функции ее сокращенной КНФ имеет полиномиальную сложность.

Доказательство. Для построения сокращенной КНФ функции $f \in WP$ необходимо найти все простые имплиценты функции *f* вида

$$x_{s_1}^{\alpha} \vee x_{s_2} \vee ... \vee x_{s_q}. \tag{4}$$

Рассмотрим вначале алгоритм поиска всех имплицент (4), в которых $\alpha = 1$. Если функция *f* имеет простую имплиценту $x_{s_1} \vee x_{s_2} \vee ... \vee x_{s_q}$, где $s_1 < ... < s_q$, то

$$f(x_{s_1} = 0,..., x_{s_q} = 0) \equiv 0, \tag{5}$$

и каждая из функций

$$f(x_{s_1} = 0,..., x_{s_{i-1}} = 0, x_{s_{i+1}} = 0,..., x_{s_q} = 0), \ i = 1, ..., q, \tag{6}$$

не равна тождественно нулю. Заметим, что ввиду (5) в СКНФ функции *f* входит элементарная имплицента

$$\bar{x}_1 \vee ... \vee \bar{x}_{s_1-1} \vee x_{s_1} \vee \bar{x}_{s_1+1} \vee ... \vee \bar{x}_{s_q-1} \vee x_{s_q} \vee \bar{x}_{s_q+1} \vee ... \vee \bar{x}_k.$$

Из этих свойств следует, что для поиска всех имплицент (4), в которых $\alpha = 1$, необходимо провести следующие процедуры. Для каждого конъюнкта $x_1^{\delta_1} \vee ... \vee x_k^{\delta_k}$ функции *f* рассматриваем функцию

$$f' \equiv f(x_{j_1} = 0,..., x_{j_q} = 0),$$

где $j_1, ..., j_q$ – номера всех координат вектора $\delta = (\delta_1, ..., \delta_k)$, равных 1. В случае, когда функция *f* имеет имплиценту $(x_{j_1} \vee ... \vee x_{j_q})$, должно выполняться (5) (сложность проверки, очевидно, полиномиальна). Для проверки простоты имплиценты необходимо рассмотреть *q* соотношений вида (6).

Для поиска простых имплицент вида $\bar{x}_{s_1} \vee x_{s_2} \vee ... \vee x_{s_q}$ следует для всякого сомножителя $x_1^{\delta_1} \vee ... \vee x_k^{\delta_k}$ из СКНФ функции *f* рассмотреть $k - q$ функций $f_i' \equiv f(x_{j_1} = 0,..., x_{j_q} = 0, x_i = 1)$, где $j_1, ..., j_q$ – номера всех координат вектора $\delta = (\delta_1, ..., \delta_k)$, равных 1, $i \in \{1, ..., k\} \setminus \{j_1, ..., j_q\}$, и способом, аналогичным описанному выше, определить: является ли функция $\bar{x}_i \vee x_{j_1} \vee ... \vee x_{j_q}$ простой элементарной имплицентой функции *f*, или нет.

Нетрудно видеть, что приведенный алгоритм с полиномиальной сложностью находит сокращенную КНФ (которая является приведенным представлением) слабо положительной функции. Утверждение доказано.

Поскольку

$$f(x_1, ..., x_k) \in WP, \text{ если и только если } f(\bar{x}_1, ..., \bar{x}_k) \in WN, \tag{7}$$

и с полиномиальной сложностью по СКНФ функции $f(x_1, ..., x_k)$ находится СКНФ функции $f(\bar{x}_1, ..., \bar{x}_k)$, то справедлив следующий результат.

Утверждение 2. Существует полиномиальный алгоритм, который по СКНФ слабо отрицательной функции *f* находит сокращенную КНФ функции *f*.

Перейдем к рассмотрению случая, когда функции задаются многочленами Жегалкина.

Утверждение 3. Задача построения по многочлену Жегалкина монотонной функции ее сокращенной (минимальной) КНФ не является полиномиальной.

Доказательство. Нетрудно показать, что для монотонной функции сокращенная КНФ совпадает с минимальной КНФ.

Для доказательства теоремы достаточно показать, что для любого полинома $p(n)$ существует натуральное k_0, такое, что при любом $k > k_0$ найдется монотонная функция $f(x_1, ..., x_k)$, для длины записи которой многочленом Жегалкина $\text{len}(f_{\text{Жег}})$ справедливо соотношение

$$\text{len}(f_{\text{КНФ}}) > p(\text{len}(f_{\text{Жег}})), \tag{8}$$

где $\text{len}(f_{\text{КНФ}})$ – длина записи сокращенной КНФ функции *f*.

Перейдем к построению монотонных функций с требуемыми свойствами. Символом $h^{(r)}(y_1,...,y_r)$ обозначим следующую монотонную функцию:

$$h^{(r)}(y_1,...,y_r) \equiv y_1 \vee ... \vee y_r \equiv \bar{y}_1 \cdot ... \cdot \bar{y}_r \oplus 1. \tag{9}$$

Поскольку запись функции $\bar{y}_1 \cdot \ldots \cdot \bar{y}_r$ многочленом Жегалкина содержит все 2^r мономов, то запись функции $h^{(r)}$ в виде многочлена Жегалкина содержит $2^r - 1$ мономов.

Рассмотрим функцию $h^{(rt)}$, которая является суперпозицией функции $h^{(r)}$ и монотонных функций

$$q^{(i)}(x_{(i-1)t+1},\ldots,x_{it}) \equiv x_{(i-1)t+1} \cdot \ldots \cdot x_{it},$$

$$h^{(rt)}(x_1,\ldots,x_{rt}) = h^{(r)}(q^{(1)}(x_1,\ldots,x_t),\ldots,q^{(r)}(x_{(r-1)t+1},\ldots,x_{rt})).$$

Длина записи функции $h^{(rt)}$ многочленом Жегалкина не более чем

$$\text{len}(h^{(rt)}_{\text{Жег}}) \le 2^r (rt)^2 . \tag{10}$$

Например, $$h^{(2,k/2)}(x_1,\ldots,x_k) \equiv x_1 \cdot \ldots \cdot x_{k/2} \oplus x_{(k/2)+1} \cdot \ldots \cdot x_k \oplus x_1 \cdot \ldots \cdot x_k,$$

где k – четное число.

Ясно, что $h^{(rt)}$ как суперпозиция монотонных функций является монотонной функцией. Кроме того, нетрудно показать, что сокращенная (а значит, и минимальная) КНФ функции $h^{(rt)}$ имеет вид

$$\bigwedge_{\substack{1\le s_1\le t,\\ t+1\le s_2\le 2t,\\ \cdots\\ (r-1)t+1\le s_r\le rt}} (x_{s_1} \vee \ldots \vee x_{s_r}).$$

Поэтому длина записи минимальной КНФ функции $h^{(rt)}$ не менее чем

$$h^{(rt)}_{\text{КНФ}} \ge t^r . \tag{11}$$

Для любого натурального k определим монотонную функцию $f(x_1, \ldots, x_k)$. Положим

$$t = \lfloor k / \log_2 k \rfloor, \quad r = \lfloor \log_2 k \rfloor .$$

Если $t = 0$ или $r = 0$, то считаем $f \equiv 0$. В случае $t \cdot r > 0$ функция $f(x_1, \ldots, x_k)$ существенно зависит от $t \cdot r$ переменных

$$f(x_1,\ldots,x_{rt},\alpha_1,\ldots,\alpha_{k-rt}) = h^{(rt)}(x_1,\ldots,x_{rt}),$$

для любых $(\alpha_1,\ldots,\alpha_{k-rt}) \in B_{k-rt}$.

Для длины записи функции f многочленом Жегалкина из соотношения (10) следует оценка сверху

$$\text{len}(f_{\text{Жег}}) \le 2^{\lfloor \log_2 k \rfloor} (rt)^2 \le 2^{\log_2 k} k^2 = k^3 . \tag{12}$$

С другой стороны, из (11) вытекает

$$\text{len}(f_{\text{КНФ}}) \ge (\lfloor k / \log_2 k \rfloor)^{\lfloor \log_2 k \rfloor} > ((k / \log_2 k) - 1)^{\log_2 k - 1}, \tag{13}$$

откуда, в частности, следует, что при $0 < \varepsilon < 1$ (например, $\varepsilon = 1/2$) справедливо следующее асимптотическое неравенство:

$$\text{len}(f_{\text{КНФ}}) \gtrsim k^{\varepsilon \cdot \log_2 k}. \tag{14}$$

Из (14) и (12) получаем справедливость (8) и теоремы в целом.

Следствие. Задача построения по многочлену Жегалкина слабо положительной функции ее сокращенной (минимальной) КНФ не является полиномиальной.

Замечание. Алгоритмы сложности вида $k^{\varepsilon \cdot \log_2 k}$ называются субэкспоненциальными алгоритмами, их сложность превосходит любой полином от k, но меньше чем 2^{k^ε} для любого $\varepsilon > 0$.

Приведем без доказательства один результат о сложности нахождения приведенного представления слабо отрицательных функций, записанных многочленом Жегалкина.

Утверждение 4. Существует алгоритм построения сокращенной КНФ любой слабо отрицательной функции $f(x_1, \ldots, x_k)$, записанной многочленом Жегалкина, сложность которого не выше чем

$$T' = \text{len}(f_{\text{Жег}})^{c \cdot \log_2 (\text{len}(f_{\text{Жег}}))} + p(\text{len}(f_{\text{КНФ}})), \tag{15}$$

где p – некоторый полином; $c > 0$ – некоторая константа.

Замечание. Таким образом, функция сложности (15) есть сумма двух слагаемых, первое слагаемое является субэкспоненциальной функцией от размера входа, а второе слагаемое – полиномиальная функция от размера выхода.

ЛИТЕРАТУРА

1. *Schaefer T.* Complexity of satisfiability problems // Proceedings of the 10 Annual ACM Symposium on Theory of Computing Machinery. 1978. P. 216 – 226.
2. *Горшков С.П.* Применение теории NP-полных задач для оценки сложности решения систем булевых уравнений // Обозрение прикл. промышл. матем., сер. дискрет. матем. 1995. Т. 2. Вып. 3. С. 325 – 398.
3. *Dowling W.F.*, *Gallier J.H.* Linear-time algorithms for testing the satisfiability of propositional Horn formulae // J. Logic Programming. 1984. No. 3. P. 267 – 284.

2008 Теоретические основы прикладной дискретной математики № 1(1)

DOI 10.17223/20710410/1/2

УДК 519.651

МОНОМИАЛЬНЫЕ ПРИБЛИЖЕНИЯ ПЛАТОВИДНЫХ ФУНКЦИЙ*

А.В. Иванов

Московский государственный институт радиотехники, электроники и автоматики

E-mail: alexiva@pochta.ru

Рассматриваются вопросы приближения платовидных булевых функций мономиальными. При исследовании свойств платовидных функций используется их представление в виде многочленов над конечным полем. Получены условия, необходимые для того, чтобы расстояние Хэмминга между векторами значений платовидной функции и любой собственной мономиальной принимало не более трех возможных значений.

Ключевые слова: *булевы функции*, *мономиальные приближения булевых функций*, *приведенное представление*, *платовидные функции.*

При решении широкого класса криптографических задач часто возникает необходимость найти приближение двоичной функции из некоторого заданного множества функций. Вероятность совпадения значений аппроксимации со значениями исходной функции при случайном равновероятном выборе аргументов является мерой качества приближения. Как правило, в роли класса приближений рассматривают множество линейных функций. Это связано с тем, что для данного множества удается решить наибольшее количество сопутствующих методу задач – найти наилучшее приближение для заданной функции, получить содержательные оценки качества приближения, описать функции, наихудшим образом приближаемые классом.

Однако в ряде случаев использование нелинейных аппроксимаций позволяет повысить эффективность решения той или иной криптоаналитической задачи. В связи с этим в последнее время предпринимаются попытки расширить класс линейных приближений с целью улучшения точности аппроксимации булевых функций при сохранении других важных характеристик класса.

В работе рассматриваются вопросы приближения платовидных булевых функций мономиальными функциями.

Используются следующие обозначения:

$\mathbb{F}_2 = \mathrm{GF}(2)$ – поле из двух элементов;

e – единица поля $\mathbb{F}_2$;

$\mathbb{F}_{2^n} = \mathrm{GF}(2^n)$ – расширение поля $\mathbb{F}_2$ натуральной степени n;

$\mathrm{tr}_t^n(\alpha) = \sum_{k=0}^{\frac{n}{t}-1} \alpha^{2^{t\cdot k}}$ – функция след из поля $\mathbb{F}_{2^n}$ в его подполе $\mathbb{F}_{2^t} = \mathrm{GF}(2^t)$ для натурального t такого, что $t|n$;

$\overline{t_1, t_2}$, где $t_1, t_2 \in Z$, – множество целых чисел, больших t_1-1 и меньших t_2+1;

$||t||$ – количество единиц в двоичной записи числа t;

$\rho_n(t)$, где $t \in Z$, – число из множества $\overline{1, 2^n-1}$, определяемое условием $\rho_n(t) \equiv t \pmod{2^n-1}$.

Далее, где это необходимо, будем отождествлять ноль и единицу поля $\mathbb{F}_2$ с целыми числами 0 и 1 соответственно. При получении основных результатов использовано представление булевых функций от n переменных в виде многочленов над полем $\mathbb{F}_{2^n}$, принимающих значения в поле $\mathbb{F}_2$. Опишем механизм получения подобного представления.

Пусть $\varphi(x_0, x_1, \ldots, x_{n-1})$ – булева функция от n переменных; $(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{n-1})$, $(\omega_0, \omega_1, \ldots, \omega_{n-1})$ – пара двойственных базисов поля $\mathbb{F}_{2^n}$ как векторного пространства над полем $\mathbb{F}_2$. Для любого набора булевых величин $x_0, x_1, \ldots, x_{n-1}$ однозначно определен элемент $x = \sum_{j=0}^{n-1} x_j \cdot \varepsilon_j$ поля $\mathbb{F}_{2^n}$. В [1] показано, что существует многочлен $\Phi(x)$ над полем $\mathbb{F}_{2^n}$, такой, что

$$\varphi(x_0, x_1, \ldots, x_{n-1}) = \mathrm{tr}_1^n\left(\Phi\left(\sum_{j=0}^{n-1} x_j \cdot \varepsilon_j\right)\right). \quad (1)$$

* Работа выполнена при поддержке гранта Президента РФ НШ №8564.2006.10.

При этом $\Phi(x)$ определен однозначно, если он имеет вид

$$\Phi(x) = \sum_{t \in M_n} c_t \cdot \xi_t \cdot x^t \,, \tag{2}$$

где M_n – набор минимальных представителей всех различных циклотомических классов множества чисел $\overline{1, 2^n - 1}$, и для каждого t:

$r(t) = \min\{k \in \mathbb{N} : t \cdot 2^k \equiv t \pmod{2^n - 1}\}$;

c_t – выбирается из поля $\mathbb{F}_{2^{r(t)}}$;

ξ_t – фиксированный элемент поля $\mathbb{F}_{2^n}$, такой, что справедливо соотношение $\mathrm{tr}^n_{r(t)}(\xi_t) = e$.

В этом случае многочлен $\Phi(x)$ вида (2) называют редуцированным многочленом, а представление (1), где многочлен $\Phi(x)$ имеет вид (2), – приведенным представлением для $\varphi(x_0, x_1, \ldots, x_{n-1})$ в базисе $(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{n-1})$.

Индекс нелинейности редуцированного многочлена $\Phi(x)$ вида (2) задается равенством

$$\mathrm{ind}\,\Phi(x) = \max\{||t||: t \in M_n,\ c_t \neq 0\}$$

и совпадает со степенью нелинейности функции $\varphi(x_0, x_1, \ldots, x_{n-1})$.

Использование подобного представления булевых функций в ряде случаев существенно упрощает решение комплекса задач по исследованию классов приближающих функций.

В качестве показателя близости приближающей функции $G(x)$ к исходной функции $F(x)$ будем использовать величину

$$\Delta(F, G) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{F(x)+G(x)}. \tag{3}$$

Если $G(x)$ – линейная функция, то $\Delta(F, G)$ есть коэффициент Уолша – Адамара функции $F(x)$. Коэффициенты Уолша – Адамара произвольной функции $F(x) : \mathbb{F}_{2^n} \to \mathbb{F}_2$ удовлетворяют равенству Парсеваля

$$\sum_{\alpha \in \mathbb{F}_{2^n}} \left(\Delta(F, \mathrm{tr}^n_1(\alpha x))\right)^2 = 2^{2n},$$

из которого следует оценка

$$\max\{|\Delta(F, \mathrm{tr}^n_1(\alpha x))| : \alpha \in \mathbb{F}_{2^n}\} \geq 2^{n/2}. \tag{4}$$

Функции, для которых неравенство (4) обращается в равенство, существуют только для четных значений n и носят название *бент-функций*.

В случае, когда число переменных нечетно, задача описания функций, плохо приближаемых линейными, значительно труднее. На пути ее решения, в частности, интерес вызывают функции, у которых число различных коэффициентов Уолша – Адамара сравнительно невелико. Из равенства Парсеваля вытекает, что если все ненулевые квадраты коэффициентов Уолша – Адамара одинаковы, то они равны 2^{2n-2s} при некотором $s \leq n/2$. Такие функции называются *платовидными порядка* $2s$ или, в случае $s = (n-1)/2$, *полубент-функциями*.

Одно из направлений задачи изучения нелинейных приближений связано с рассмотрением в качестве приближающего множества класса так называемых мономиальных функций, то есть таких, у которых редуцированный многочлен приведенного представления состоит из одного монома (см. [1]). В работе [5] приведены примеры функций от четного числа переменных, наихудшим образом приближаемых собственными мономиальными (редуцированный многочлен которых задает подстановку на $\mathbb{F}_{2^n}$). От авторов они получили название «гипербент-функций». Позднее в работе [6] была предпринята попытка использовать аппарат построения гипербент-функций для случая нечетного n.

В рамках данной работы мы рассмотрим вопрос о существовании функций, для которых при некотором s выполнено:

$$(\Delta(F, \mathrm{tr}^n_1(\alpha x^\delta)))^2 \in \{0, 2^{2n-2s}\} \text{ для любых } \alpha \in \mathbb{F}_{2^n} \text{ и } \delta \in \overline{1, 2^n - 1},\ (\delta, 2^n - 1) = 1. \tag{5}$$

Замечание 1. Известно (см., например, [3]), что если для всех $\alpha \in \mathbb{F}_{2^n}$

$$|\Delta(F, \mathrm{tr}^n_1(\alpha x))| \equiv 0 \pmod{2^w},$$

то $\deg F \leq n - w + 1$. Следовательно, степень платовидной функции порядка $2s$ не превосходит $s + 1$.

Теорема. Необходимыми условиями существования платовидных функций порядка $2s$ от нечетного числа переменных n, для которых выполняется условие (5), являются:

1) $s = (n-1)/2$;

2) существуют натуральные σ, делящие $2^n - 1$, такие, что для любого δ, взаимно простого с $2^n - 1$,

$$||\rho_n(\sigma\delta)|| \in \{(n-1)/2, (n+1)/2\}.$$

Доказательство. В доказательстве теоремы применим рассуждения, аналогичные использованным в [1].

Пусть $F(x):\mathbb{F}_{2^n}\to\mathbb{F}_2$ – платовидная порядка $2s$ функция от n переменных. Выберем натуральное δ взаимно простым с 2^n-1. Тогда существует натуральное $\eta=\eta(\delta)$, такое, что $\delta\eta\equiv 1 \pmod{2^n-1}$.

Рассмотрим функцию $F_\eta(x):\mathbb{F}_{2^n}\to\mathbb{F}_2$, определяемую равенством $F_\eta(x) = F(x^\eta)$ для любого x из поля $\mathbb{F}_{2^n}$. Так как

$$\Delta\left(F,\mathrm{tr}_1^n\left(\alpha x^\delta\right)\right)=\sum_{x\in\mathbb{F}_{2^n}}(-1)^{F(x)+\mathrm{tr}_1^n\left(\alpha x^\delta\right)}=\sum_{x\in\mathbb{F}_{2^n}}(-1)^{F\left(x^\eta\right)+\mathrm{tr}_1^n\left(\alpha x^{\delta\eta}\right)}=\Delta\left(F_\eta,\mathrm{tr}_1^n\left(\alpha x\right)\right),$$

то выполнение для всех $\alpha\in\mathbb{F}_{2^n}$ условия

$$(\Delta(F,\mathrm{tr}_1^n(\alpha x^\delta)))^2\in\{0,2^{2n-2s}\}$$

равносильно тому, что функция $F_\eta(x)$ также является платовидной порядка $2s$.

Пусть теперь

$$\Phi(x)=\sum_{t\in M_n}c_t\cdot\xi_t\cdot x^t$$

и

$$\Phi_\eta(x)=\sum_{t\in M_n}c'_t\cdot\xi_t\cdot x^t$$

– редуцированные многочлены приведенных представлений функций $F(x)$ и $F_\eta(x)$ соответственно. Согласно [1],

$$\mathrm{ind}\,\Phi_\eta(x)=\max\{||\rho_n(t\eta)||:t\in M_n,c_t\neq 0\}. \tag{6}$$

В соответствии с замечанием 1 для выполнения условия (5) необходимо

$$\mathrm{ind}\,\Phi(x)\le s+1 \quad\text{и}\quad \mathrm{ind}\,\Phi_\eta(x)\le s+1. \tag{7}$$

Перебирая все $\delta\in\overline{1,2^n-1}$, такие, что $(\delta,2^n-1)=1$, и строя для них натуральные $\eta=\eta(\delta)$ со свойством

$$\delta\eta\equiv 1 \pmod{2^n-1},$$

получим опять все множество натуральных чисел, меньших числа 2^n-1 и взаимно простых с ним. Тогда из выражений (6) и (7) следует, что (5) имеет место, только если

$||\rho_n(t\eta)||\le s+1$ для всех $t\in M_n$ и всех $\eta\in\overline{1,2^n-1}$, таких, что $c_t\neq 0$ и $(\eta,2^n-1)=1$.

Зафиксируем элемент η с указанными свойствами и рассмотрим $\eta'=2^n-1-\eta$. Очевидно, что $(\eta',2^n-1)=1$. Тогда выполнение (5) требует

$$||\rho_n(t\eta)||\le s+1 \text{ и } ||\rho_n(t\eta')||\le s+1 \text{ для всех } t\in M_n, \text{ таких, что } c_t\neq 0. \tag{8}$$

При этом

$$\rho_n(t\eta')=2^n-1-\rho_n(t\eta)$$

и

$$||\rho_n(t\eta')||=n-||\rho_n(t\eta)||. \tag{9}$$

Из (9) следует, что условие (8) влечет

$$||\rho_n(t\eta)||\le s+1,\ n-||\rho_n(t\eta)||\le s+1 \text{ для всех } t\in M_n, \text{ таких, что } c_t\neq 0,$$

или, что то же,

$$n-s-1\le||\rho_n(t\eta)||\le s+1 \text{ для всех } t\in M_n, \text{ таких, что } c_t\neq 0.$$

Последнее выполнено только тогда, когда

$$s\ge(n-2)/2. \tag{10}$$

Так как порядок платовидной функции от нечетного числа переменных n не может превышать $(n-1)/2$, то из (10) следует

$$s=(n-1)/2. \tag{11}$$

Таким образом, необходимым условием выполнения (5) является

$$||\rho_n(t\eta)||\in\{(n-1)/2,(n+1)/2\} \text{ для всех } t\in M_n,\eta\in\overline{1,2^n-1}, \text{ таких, что } c_t\neq 0,(\eta,2^n-1)=1. \tag{12}$$

Очевидно, что для любого $t\in M_n$ существует σ_t такой, что

$$(\sigma_t,2^n-1)=1$$

и

$$\rho_n(t\sigma_t)=(t,2^n-1).$$

Следовательно, (12) равносильно

$$||\rho_n(\sigma_t\eta)||\in\{(n-1)/2,(n+1)/2\} \text{ для всех } t\in M_n,\eta\in\overline{1,2^n-1}, \text{ таких, что } c_t\neq 0,(\eta,2^n-1)=1. \tag{13}$$

(11) и (13) в совокупности доказывают утверждение теоремы.

Следствие. Если число $2^n - 1$ простое, то функций от n переменных, удовлетворяющих условию (5), не существует.

Утверждение 1. Пусть n нечетно, $G(y) : \mathbb{F}_{2^n} \to \mathbb{F}_2$, $t|(2^n - 1)$ и функция $F(x) : \mathbb{F}_{2^n} \to \mathbb{F}_2$ определена условием $F(x) = G(x^t)$ для каждого $x \in \mathbb{F}_{2^n}$. Тогда если $t > 1$, то $F(x)$ не может являться платовидной порядка n–1.

Доказательство. Очевидно, что вес платовидной функции порядка n–1 может принимать одно из трех значений:

$$2^{n-1},\ 2^{n-1} - 2^{(n-1)/2},\ 2^{n-1} + 2^{(n-1)/2}.$$

С другой стороны, если θ – примитивный элемент поля $\mathbb{F}_{2^n}$, то период последовательности

$$(F(\theta^{ti}) : i \in \mathbb{N})$$

делит $(2^n - 1)/t$. Следовательно, вес функции $F(x)$ кратен t.

Рассмотрим три возможных случая.

1) Так как $(2^{n-1}, 2^n - 1) = 1$, то $F(x)$ имеет вес 2^{n-1}, только если $t = 1$.

2) Так как $2^n - 1$ нечетно, то

$$(2^{n-1} - 2^{(n-1)/2}, 2^n - 1) = (2^{(n-1)/2}(2^{(n-1)/2} - 1), 2^n - 1) = (2^{(n-1)/2} - 1, 2^n - 1).$$

Согласно [2],

$$(2^{(n-1)/2} - 1, 2^n - 1) = 2^{((n-1)/2,\ n)} - 1,$$

но $(n-1)/2$ и n, очевидно, взаимно просты. Значит,

$$(2^{n-1} - 2^{(n-1)/2}, 2^n - 1) = 2^{((n-1)/2,\ n)} - 1 = 1$$

и $F(x)$ имеет вес $2^{n-1} - 2^{(n-1)/2}$, только если $t = 1$.

3) Так как $2^n - 1$ нечетно и $(2^{(n-1)/2} - 1, 2^{(n-1)/2} + 1) = 1$, то

$$(2^{n-1} + 2^{(n-1)/2}, 2^n - 1) = (2^{(n-1)/2}(2^{(n-1)/2} + 1), 2^n - 1) = (2^{(n-1)/2} + 1, 2^n - 1) = ((2^{n-1} - 1, 2^n - 1)/(2^{(n-1)/2} - 1, 2^n - 1)).$$

Из п. 2 следует, что

$$((2^{n-1} - 1, 2^n - 1)/(2^{(n-1)/2} - 1, 2^n - 1)) = (2^{(n-1,\ n)} - 1).$$

Так как $(n-1, n) = 1$, то

$$(2^{n-1} + 2^{(n-1)/2}, 2^n - 1) = 1$$

и $F(x)$ имеет вес $2^{n-1} + 2^{(n-1)/2}$, только если $t = 1$.

Таким образом, из пп. 1, 2 и 3 следует, что при $t > 1$ не существует платовидных функций порядка n–1, представимых в указанном виде.

Утверждение 2. Если для некоторого нечетного n, большего пяти, мощность множества

$$W_n = \{\sigma \in \overline{1, 2^n - 1} : \sigma | (2^n - 1),\ ||\rho_n(\sigma\delta) \pmod{2^n - 1}|| \in \{(n-1)/2, (n+1)/2\}$$

$$\text{для всех } \delta \in \overline{1, 2^n - 1},\ \text{таких, что } (\delta, 2^n - 1) = 1\},$$

не превосходит двух, то функций от n переменных, удовлетворяющих условию (5) для какого-либо значения s, не существует.

Доказательство. Если множество W_n пусто, то результат утверждения следует непосредственно из теоремы.

Если W_n состоит из единственного элемента σ, то редуцированный многочлен приведенного представления функции $F(x)$, удовлетворяющей условию (5), должен (см. доказательство теоремы) иметь вид

$$\Phi(x) = \sum_{\substack{\eta \in \overline{1, 2^n - 1}: \\ (\eta, 2^n - 1) = 1}} c_\eta \left(x^\sigma\right)^\eta.$$

Следовательно, функция $F(x)$ представима в форме $F(x) = G(x^\sigma)$ и, согласно утверждению 1, не может являться платовидной порядка n–1.

Пусть $W_n = \{\sigma', \sigma''\}$. В этом случае редуцированный многочлен приведенного представления функции $F(x)$, удовлетворяющей условию (5), должен иметь вид

$$\Phi(x) = \sum_{\substack{\eta \in \overline{1, 2^n - 1}: \\ (\eta, 2^n - 1) = 1}} \left(c'_\eta \left(x^{\sigma'}\right)^\eta + c''_\eta \left(x^{\sigma''}\right)^\eta\right).$$

Следовательно, $F(x)$ представима в форме

$$F(x) = G(x^{(\sigma', \sigma'')}). \tag{14}$$

Покажем, что $(\sigma', \sigma'') > 1$. Пусть это не так. Тогда σ' и σ'' – взаимно простые делители числа $2^n - 1$. Получим

$$\sigma' \cdot \sigma'' \mid 2^n - 1. \tag{15}$$

С другой стороны, так как двоичный вес этих чисел не меньше $(n-1)/2$ и $(2^{(n-1)/2} - 1, 2^n - 1) = 1$, то, не ограничивая общности,

$$\sigma' \geq 3 \cdot 2^{(n-3)/2} - 1$$

и

$$\sigma'' \geq 7 \cdot 2^{(n-5)/2} - 1.$$

В условиях утверждения это влечет

$$\sigma' \cdot \sigma'' \geq (3 \cdot 2^{(n-3)/2} - 1)(7 \cdot 2^{(n-5)/2} - 1) = 42 \cdot 2^{(n-5)} - 13 \cdot 2^{(n-5)/2} + 1 = 2^n - 1 + (10 \cdot 2^{(n-5)} - 13 \cdot 2^{(n-5)/2} + 2) > 2^n - 1.$$

Но последнее противоречит (15). Таким образом, $(\sigma', \sigma'') > 1$. Тогда из (14) и результата утверждения 1 следует, что $F(x)$ не может являться платовидной порядка $n-1$.

Утверждение доказано.

Замечание 2. Из теоремы следует, что для существования платовидных функций порядка $n-1$ от n переменных, удовлетворяющих условию (5), необходимо наличие делителей σ числа $2^n - 1$, таких, что для любого δ, взаимно простого с $2^n - 1$,

$$||\rho_n(\sigma\delta)|| \in \{(n-1)/2, (n+1)/2\}.$$

По результатам эксперимента для $5 \leq n \leq 33$ числа с указанным свойством обнаружены только при

$$n = 21 \ (\sigma = 42799)$$

и

$$n = 29 \ (\sigma = 256999).$$

Таким образом, опираясь на данные эксперимента и утверждение 2, можно заключить, что для $n \leq 33$ не существует функций, удовлетворяющих условию (5).

ЛИТЕРАТУРА

1. *Кузьмин А.С.*, *Марков В.Т.*, *Нечаев А.А.*, *Шишков А.Б.* Приближение булевых функций мономиальными // Дискр. мат. 2006. Т. 18. № 1. С. 9 – 29.
2. *Лидл Р.*, *Нидеррайтер Г.* Конечные поля. Т. 1, 2. М.: Мир, 1988.
3. *Логачев О.А.*, *Сальников А.А.*, *Ященко В.В.* Булевы функции в теории кодирования и криптологии. М.: МНЦМО, 2004.
4. *Мак-Вильямс Ф. Дж.*, *Слоэн Н. Дж.* Теория кодов, исправляющих ошибки. М.: Связь, 1979.
5. *Youssef A.M.*, *Gong G.* Hyper-bent functions // Proceedings of Advances in Cryptology: EUROCRYPT'2001. Lect. Notes in Comp. Sci. New York: Springer Verlag, 2001. V. 2045. P. 406 – 419.
6. *Youssef A.M.*, *Gong G.* Boolean functions with large distance to all bijective monomials: N odd case // Proceedings of the Eighth Annual Workshop on Selected Areas in Cryptography, Toronto, August 16 – 18, 2001.

2008 Теоретические основы прикладной дискретной математики № 1(1)

DOI 10.17223/20710410/1/3
УДК 519.1

КОДЫ, КОМПОЗИЦИИ И РЕШЕТКИ

А.М. Кутьин

Сибирский федеральный университет, г. Красноярск

E-mail: maglinetc@mail.ru

Обсуждаются вопросы теории кодирования, связанные с ее центральной проблемой. Рассмотрена основная модель теории кодирования и криптографии – дистрибутивная решетка, исследованы ее связи с композициями чисел и *c*-матроидами. Построены решеточные коды. Указаны границы кодов в метрике Хемминга.

Ключевые слова: *код*, *композиции чисел*, *решетка*, *метрика*, *s-множество*, *c-матроид*

Теория кодирования и криптографии призвана решить относящиеся к ней задачи общей проблемы безошибочной передачи данных с обеспечением их защиты от несанкционированного доступа. Одной из теоретических основ их решения, как показано ниже, является теория решеток и s-множеств [1 – 7].

Напомним, что решетка как алгебра обозначается обычно $L = < M; +, \cdot >$ или $L = < M; \vee, \wedge >$, $M \neq \varnothing$.

Центральная проблема теории кодирования имеет немалое число формулировок. Приведем лишь две из них. При заданных n и d найти $A(n, d)$, равное максимальному числу кодовых слов в равномерном (n, M, d)-коде, где M – объем кода, см. [8]. Другая формулировка [9]: найти равномерные коды с большим $R = k/n$ для эффективности, так как R – скорость, и большим d для надежности: чем больше d, тем больше ошибок можно исправить (здесь k – число информационных символов). Мы предпочитаем следующую формулировку.

Найти аналитическую формулу для функции $A(m, l, d)$, где $A(m, l, d)$ – максимальная мощность кодов из слов v равной длины $l(v) = n$ над алфавитом мощности m с расстоянием ρ между кодовыми словами не менее наперёд заданного d; в отсутствие точной формулы найти неулучшаемые верхние и нижние границы.

1. Кодирование и *L*-кодирование

Рассмотрим сначала равномерные коды (кратко Р-коды). Возьмем алфавит $A = \{a_0, \ldots, a_m\}$ и построим множество W всех слов v равной длины $l(v) = n$ над A; получаем простой равномерный код (ПРК, или ПР-код) над A. Слово $u = x_1 \ldots x_n$ можно рассматривать и как вектор $\boldsymbol{u} = <x_1, \ldots, x_n>$ конечного n-мерного пространства, $x_i \in A$, $i = 1, \ldots, n$. Поэтому можно говорить о координатах или компонентах в векторах (словах) и символах в словах (векторах). Исходным фактом теории L-кодирования, который объясняет ее название, является следующая теорема 1.

Теорема 1. Любое множество W слов v равной длины $l(v) = n$, в частности простой равномерный код, над линейно упорядоченным алфавитом $A = \{a_0, \ldots, a_m\}$ объема $m+1$ при векторном отношении порядка $\leq$ на W, индуцированном порядком на A, является конечной дистрибутивной самодвойственной решеткой $L_A = \langle W, \leq \rangle$, порожденной c-элементами вида $c_{i\,j} = (a_0, \ldots, a_0, a_j, a_0, \ldots, a_0)$, где a_j стоит на j-м месте. При этом:

1) L_A – градуированная решетка, число уровней которой равно $nm+1$, представляет собой прямое произведение цепей C_i равной длины $l(C_i)$ числом n: $L_A = \Pi_1^n C_i$, каждая из которых состоит только из c-элементов указанного вида, а всякий c-элемент принадлежит соответствующей цепи C_i, и только ей;

2) число n всех цепей C_i равно $n = l(v)$ – длине слова v (равносильно – длине кода), а длина $l(C_i)$ любой цепи C_i равна $m = l(C_i)$, где $m+1 = |A|$ – объем алфавита A, причем $l(L_A) = l(\Pi_1^n C_i) = \Sigma_1^n\, l(\mathrm{C}_i) = nm$;

3) число всех c-элементов равно $nm+1$, причем $c_{ik} \wedge c_{jt} = \hat{0}$ при $i \neq j$, а любое слово v Р-кода представимо единственной суммой $v = \vee\, c_{km}$ c-элементов, в этом слове содержащихся: $v \geq c_{km}$ для всех c_{km} из суммы $\vee\, c_{km}$;

4) множество всех c-элементов есть у-подмножество в решетке L_A всех $\vee$-неприводимых элементов из L_A, являющееся $\wedge$-полурешеткой ранга m и c-матроидом, а при добавлении к ним всех элементов из L_A рангов, соответствующих рангам c-элементов, оказывается также и суперматроидом.

Обратно, любая конечная решетка L, разложимая в прямое произведение цепей $L = \Pi_1^n C_i$ равной длины m, являясь самодвойственной дистрибутивной решеткой, порожденной $\vee$-неприводимыми элементами, изоморфна решетке L_A слов равной длины n над алфавитом $A = \{0, 1, \ldots, m\}$. ■

В этом контексте удобно L_A обозначать через $L_{\text{ПРК}}(m, n)$, $m+1 = |A|$ или через $L_{\text{ПРК}}(n)$ при фиксированном m, а в c_{ik} удалить индекс k, так что далее c_i обозначает просто некоторый c-элемент.

Определим ранг-метрику ρ_r: $\rho_r(v, u) = r(v \vee u) - r(v \wedge u)$, где $v, u \in L_A$, $r(x)$ – ранговая функция на L_A (по сути, $r(x)$ – это высота элемента x в L_A), и H-метрику ρ_H: пусть $w_H(x)$ – вес Хемминга вектора x, тогда $\rho_H(x, y) = w_H(x - y)$ выражает число позиций, в которых отличаются векторы x и y; значит, в любой L_A всегда $\rho_H \le n$, где n – длина Р-кода, а $\rho_H = n$ – максимальное значение функции ρ_H.

Следствие 1. Группа автоморфизмов решетки $L_{\text{ПРК}}(n)$ изоморфна симметрической группе S_n перестановок, где n – длина кода над односортным алфавитом A, а используя перестановки из S_n и/или из симметрической группы $S(A)$ перестановок на A, и/или элементы из группы антиизоморфизмов решетки $L_{\text{ПРК}}(n)$, можно все слова Р-кода преобразовать в другие, тем самым шифруя закодированное сообщение.

Решетка L_A при любом алфавите A допускает ранг-метрику ρ_r и H-метрику ρ_H. Пара $<L_A, \rho_r>$, как и пара $<L_A, \rho_H>$, является дискретным метрическим пространством при любом алфавите A. ■

Теорема 1 позволяет изучать коды независимо от их происхождения: созданы ли они над полями, или другими алгебрами, или являются геометрическими кодами и т.д. Например, на рис. 1 представлен ПРК над кольцом классов вычетов по модулю 3, а на рис. 2 в [1] – ПРК над симметрической полугруппой перестановок S_n.

Тем самым теория L-кодирования инвариантна относительно источников происхождения кодов, и одновременно она позволяет использовать все факты, полученные иным путем, так как все, что определено и доказано для множества слов W равной длины над любым алфавитом A и его подмножества кодовых слов, конечно же, определено и доказано для L_A. Поэтому здесь и не приведены известные факты теории кодирования, кроме тех, что понадобятся далее для целей изложения.

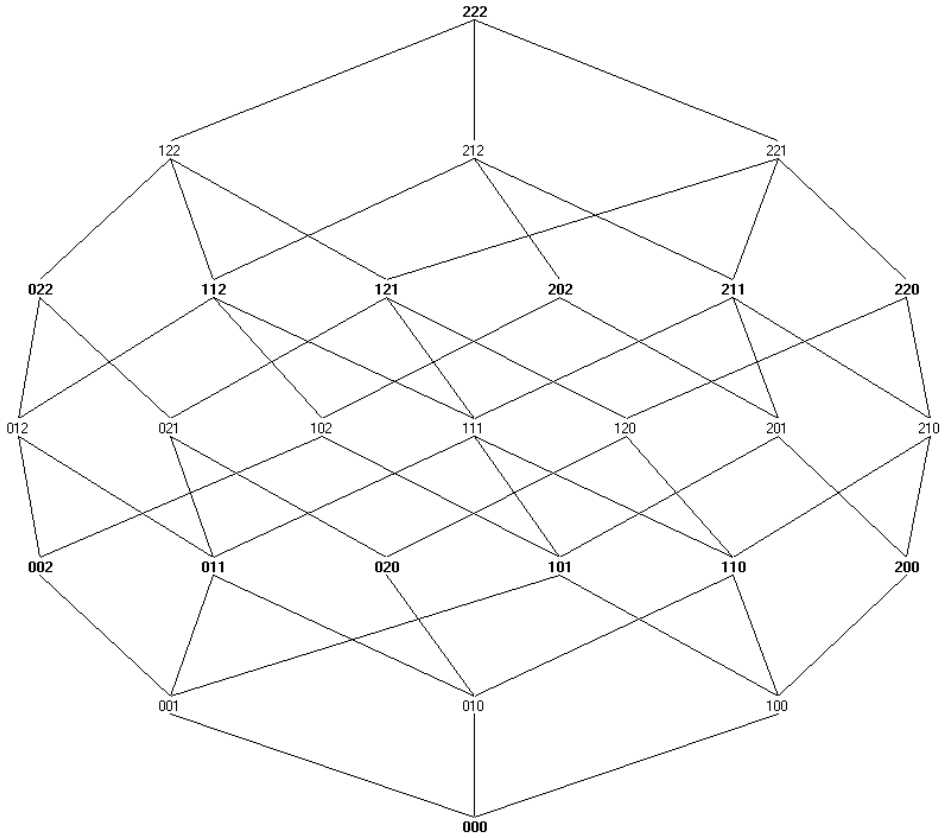


Рис. 1. Решётка слов длины 3 над алфавитом $A = \{0, 1, 2\}$. Слова кода кусочно-постоянного веса (либо 6, либо 4, либо 2, либо 0) с минимальным кодовым расстоянием $d_0 = 2$ выделены

Как видим, при $A = \{0, 1\}$ получаем булеву решетку, называемую в случае двоичных Р-кодов еще и n-мерным кубом, так что $L_{\text{ПРК}}(m, n)$ можно назвать n-мерным параллелепипедом, так как цепи C_i в общем случае можно брать различной длины, см. [1]. Булеву подрешетку двоичного ПР-кода с 2^3 словами длины 3 над алфавитом $A = \{0, 1\}$ с наименьшим в ней элементом-словом 000 и наибольшим в ней элементом-словом 111 см. на рис. 1, так что и в общем случае (при соблюдении естественного условия на длины кодов) вложение $A_1 \subseteq A_2$ алфавита A_1 в A_2 влечет вложение кода длины n над A_1 в код длины $n + k$ над A_2, $k \ge 0$, в частности, при $k = 0$ ПР-код длины n над $A_1 = \{1, \ldots, m\} \subseteq \{0, 1, \ldots, m\} = A_2$ есть подрешетка с наименьшим в ней элементом-словом 1…1 и наибольшим в ней элементом-словом $m…m$; точнее, эта подрешетка есть интервал $I = [1…1, m…m]$, изоморфный $L_{\text{ПРК}}(n, m)$ над алфавитом $A_1 = \{1, \ldots, m\}$: $I \cong L_{\text{ПРК}}(n, m)$.

Решетка на рис. 1 – это одновременно

- решетка композиций чисел от 0 до $nm = 9$ с ограничениями: 1) все $a_i \le m$ в сумме $\sum a_i = j$, здесь j – это j-уровень, $0 \le j \le nm$ (соответственно $3 \le j \le nm$); 2) число слагаемых в $\sum a_i = j$ равно n;
- решетка конечного линейного векторного пространства размерности 3, оси которого и есть цепи C_i, $i = 1, 2, 3$;
- решетка кольца (над кольцом классов вычетов по модулю 3) с покомпонентными сложением и умножением;
- решетка делимости, называемая также решеткой НОД-НОК или же решеткой упорядоченных разложений в произведение степеней простых чисел p_i^{ki}, так как компоненты элементов решетки, изображенные на рис. 1, можно интерпретировать как показатели степеней k_i этих чисел;
- решетка всех размещений с повторениями, так что их число в данном случае равно $m^n = 3^3 = 27$.

Ясно, что любая решетка $L_{\text{ПРК}}(m, n)$ обладает всеми только что перечисленными свойствами (указанные в примере конкретные значения n и m относятся именно к нему, а в общем случае их из свойств нужно, конечно, удалить).

Р-код, образуя у-подмножество в решетке $L_{\text{ПРК}}(m, n)$, не образует, вообще говоря, подрешетку в ней. Однако верно

Предложение 1. Решеточные коды, т.е. коды, являющиеся подрешетками решетки $L_{\text{ПРК}}(n)$, существуют, более того, существуют корректирующие решеточные коды.

Доказательство. Простые решеточные коды существуют – ими являются все ПРК. Покажем, что существуют корректирующие решеточные коды в любой $L_{\text{ПРК}}(m, n)$ с ранг-метрикой. Возьмем лишь те c-элементы c_i, c_j решетки $L_{\text{ПРК}}(n)$, которые находятся друг от друга на расстоянии $\rho_r(c_i, c_j) \ge d_0$ – наименьшего подходящего кодового расстояния. Найдем все элементы вида $v_k = \vee c_i$, затем все элементы вида $u_t = \wedge v_k$, потом вида $w_s = \vee u_t$ и т.д. По окончании построения множества всех таких элементов получим решеточный Р-код. В этом Р-коде для любых его кодовых слов $v = \vee c_i$ и $u = \vee c_j$ выполняется $\rho_r(v, u) \ge d_0$, потому что имеет место следующая лемма 1.

Лемма 1. Пусть $v = \vee c_i$, $u = \vee c_j$ и $\rho_r(c_i, c_j) \ge d_0$ хоть для одной пары c_i, c_j. Тогда $\rho_r(v, u) \ge \rho_r(c_i, c_j) \ge d_0$. ■

Полученный в ходе доказательства предложения 1 Р-код – это корректирующий код при $d_0 \ge 2$, так как для любых его кодовых слов $v \ne u$ выполняется $\rho_r(v, u) \ge d_0$. Например, взяв в $L_{\text{ПРК}}(2, 3)$ элементы 200, 020, 002 (см. рис. 1) и найдя все $\wedge c_i$, $\vee c_i$ и т.д., получим решеточный Р-код 000, 200, 020, 002, 220, 202, 022, 222, для любых кодовых слов v и u которого выполняется $\rho_r(v, u) \ge d_0 = 2$, так что этот код обнаруживает одну ошибку.

Все решеточные корректирующие коды того же типа для любой $L_{\text{ПРК}}(m, n)$, $m \ge 2$, с ранг-метрикой суть 00…0, m0…0, 0m…0, 0…0m, mm0…0, …, $mm...m$, а так как $\rho_r(v, u) \ge d_0 = m$, то при $m \ge 3$ эти коды не только обнаруживают, но и исправляют ошибки. Решеточные коды этого типа можно использовать для шифрования, например, двоичных ПР-кодов типа 00…0, 10…0, 01…0, 0…01, 110…0, …, 11...1, превращая их в коды m0…0, 0m…0, 0…0m, mm0…0, …, $mm...m$; при этом, хотя решетки этих кодов изоморфны, они при $m \ge 2$ существенно различны: полученный код – это корректирующий код, а исходный двоичный код таковым, конечно, не является.

Если же взять подходящие вложения $L_{\text{ПРК}}(m, n)$ в $L_{\text{ПРК}}(m+k, n)$ или в $L_{\text{ПРК}}(m+k, n+t)$, $k, t \ge 1$, то, исходя из кодов указанного типа в $L_{\text{ПРК}}(m, n)$, можно построить решеточные коды и другого типа соответственно в $L_{\text{ПРК}}(m+k, n)$ и $L_{\text{ПРК}}(m+k, n+t)$.

2. *L*-модель равномерных кодов

С учетом сказанного в п. 1 введем следующую L-модель ПР- и Р-кодов.

Основное определение. Множество всех элементов решетки $L(m, n) = \Pi_1^n C_i$, $l(C_i) = m$, назовем ПР-кодом, считая, что $L(m, n)$ есть $L_{\text{ПРК}}(m, n)$, а любое непустое подмножество элементов решетки $L(m, n) = L_{\text{ПРК}}(m, n)$ назовем Р-кодом.

Цепи и антицепи. **Уровневые числа**. Наиболее удобной решеткой $L_{\text{ПРК}}(m, n)$ для дальнейшего исследования является решетка, построенная над алфавитом $A = \{0, 1, \ldots, m\}$, так как тогда любое целое j, определяющее j-уровень этой решетки, равно весу $\omega(v)$ любого слова v этого j-уровня: $j = \omega(v)$, $0 \le j \le nm$.

Один из основных классов кодов – коды с постоянным весом строятся на основе групп перестановок. Взяв $L_{\text{ПРК}}(m, n)$, можно построить коды с кусочно-постоянным весом, пример одного из которых мы и предъявляем на рис. 1. Ясно, что коды с постоянным весом – частный случай кодов с кусочно-постоянным весом.

В дополнение к аналогам $h(\Pi_1^k P_i) = \Sigma_1^k h(P_i)$ и $l(\Pi_1^k C_i) = \Sigma_1^k l(C_i)$, см. [1], основного свойства логарифмической функции: $\ln(x \cdot y) = \ln(x) + \ln(y)$ приведем также следующий аналог этого свойства. Пусть $Ach(j)$ – антицепь, являющаяся j-уровнем решетки $L = \Pi_{i=1}^n C_i$ и $ach(j) = |Ach(j)| = w_j$ – уровневые числа (числа Уитни).

Лемма 2. При $0 \le j \le m$ и $n \ge 2$ для любой антицепи $Ach(j)$ j-уровня решетки $L = \Pi_{i=1}^n C_i$, $l(C_i) = m$, $i = 1, \ldots, n$, справедливо

$$ach(j, \Pi_{i=1}^n C_i) = \Sigma_{t=0}^j ach(t, \Pi_{i=1}^{n-1} C_i), \text{ равносильно } w_j(\Pi_{i=1}^n C_i) = \Sigma_{t=0}^j w_t(\Pi_{i=1}^{n-1} C_i). \blacksquare$$

Объемы шаров и границы. Пусть $A(n, m, d_0)$ – равномерный код (подмножество кодовых слов множества W всех слов равной длины $l = n$ над алфавитом объема $m + 1$), любые два слова которого находятся на расстоянии $\rho(v, w) \le d_0$ – минимального кодового расстояния. Пусть далее $Sh(r, z):=\{x: \rho(x, z) \le r\}$ – шар радиуса $r \ge 0$ и с центром z, z – некоторое слово, например кодовое, внутренность шара $Int(Sh(r, z)):= \{x: \rho(x, z) < r, r \ge 1\}$, сфера $Sf(r, z):= Sh(r, z) \setminus Int(Sh(r, z)) = \{x: \rho(x, z) = r\}$ и $Vsh(r, z):=|Sh(r, z)|$ – объем шара. Тогда

$$Vsh(r, z) = |Sf(r, z) \cup Int(Sh(r, z))| = |Sf(r, z)| + |Int(Sh(r, z))| \text{ при } r \ge 1.$$

При $r = 0$ считаем по определению, что шар $Sh(0, z)$ совпадает со своей внутренностью, а площадь его сферы равна 0.

Ясно, что минимальный объем шара $Sh(r, z)$ в $L(n, m)$ при $r \ge 1$ и кодовом слове z не может быть меньше 2, так как он должен содержать хотя бы 2 элемента, а чтобы код мог обнаруживать хоть одну ошибку, нужно, чтобы $d_0 \ge 2$, а тогда $r \le d_0/2$.

Объемы шаров в ранг-метрике одного и того же радиуса r, вообще говоря, различны (см. рис. 1), поэтому говорим о минимальном значении объемов шаров $Vsh_{\min}(r)$, максимальном $Vsh_{\max}(r)$ и среднем значении $Vsh_{sr}(r)$ объемов шаров. Тогда в общем случае непосредственно получаем следующие неравенства и границы

$$m^n / Vsh_{\max}(r) \le m^n / Vsh_{sr} \le m^n / Vsh_{\min}, \ (m+1)^n / Vsh_{\max}(r) \le \max|A(n,m,d_0)| \le \lceil (m+1)^n / d_0 \rceil. \blacksquare$$

Здесь и ниже $m + 1$ – объем алфавита $A = \{0, \ldots, m\}$ и $\lceil\ \rceil$ – целая часть числа сверху, например, $\lceil 25:2 \rceil = 13$.

В частном случае двоичного алфавита $A = \{0, 1\}$, $m = 1$, имеем

$$2^n / Vsh_{\max}(r) \le \max|A(n, m, d_0)| \le \lceil 2^n / d_0 \rceil.$$

Очевидно, что точное значение $\max|A(n, m, d_0)|$ находится в ряду чисел

$$\lceil m^n / Vsh_{\max}(r) \rceil, 1+\lceil m^n / Vsh_{\max}(r) \rceil, 2+\lceil m^n / Vsh_{\max}(r) \rceil, \ldots, \lceil m^n / d_0 \rceil.$$

Оказалось, что хотя объемы шаров в ранг-метрике одного и того же радиуса r, вообще говоря, различны, существуют максимальные Р-коды объема $\max|A(n, m, d_0)| = \lceil m^n : d_0 \rceil$, так что указанная верхняя граница $\lceil m^n : d_0 \rceil$ достигается. Приведем из них простейший код объема $\max|A(2, 5, 2)| = 13$ с $d_0 = 2$, обнаруживающий любую одну ошибку: 00, 11, 02, 20, 22, 13, 31, 33, 04, 40, 24, 42, 44, где взят алфавит {0, 1, 2, 3, 4}, а каждый шар имеет радиус $r = 1$; это же значение объема кода получаем согласно формуле $\max|A(n, m, d_0)| = \lceil m^n : d_0 \rceil$ при $n = 2$, $m = 5$ и $d_0 = 2$.

Так как в случае H-метрики длина Р-кода n – это максимальное значение функции ρ_H, то в этом случае шары, центры которых – кодовые слова, а внутренности не пересекаются, есть смысл брать лишь с радиусом r для $0 \le r < n$.

Если на $L(n, m)$ задана H-метрика, то площади сфер одного и того же радиуса r равны, а шары одного и того же радиуса r равнообъёмны, так что при фиксированном радиусе r соответственно имеем

$$\text{площадь любой сферы } |Sf(r, z)| = C_n^r m^r, \text{ объём любого шара } Vsh(r, z) = 1 + \sum_{ri=1}^{r} C_n^{ri} m^{ri}.$$

Поэтому, в случае H-метрики при $m = 1$ с учётом равнообъёмности шаров одного того же радиуса r получаем

$$2^n / (1 + \sum_{ri=1}^{r} C_n^{ri}) \le \max|A(n, m, d_0)| \le \lceil 2^n / d_0 \rceil.$$

***L*(*m*,*n*) как решетки композиций чисел от 0 до *nm* с ограничениями.** Как известно [10, Гл.1, п.2С], упорядочение множества разложений целого положительного числа n в суммы $n_1 + n_2 + \ldots + n_k$ целых положительных неупорядоченных слагаемых (так что, например, 1+2 = 2+1) приводит к у-множеству $D(n)$ – *доминирующему порядку*. Если же и слагаемые в сумме также упорядочены, так что, например, $1+2 \ne 2+1$, то получаем решетку $D_{\le}(n)$ композиций числа n (см. диаграммы $D(5)$ и $D_{\le}(5)$ на рис. 2).

Утверждение 1.

1. $D(n)$ является ранжированной модулярной мультирешеткой [3, 5, 11].

2. $D_{\le}(n)$ – булева решетка, точнее: $D_{\le}(n) \cong Bool(n–1)$.

3. Каждому элементу вида $n_1 + n_2 + \ldots + n_k$ из $D(n)$ соответствует свой класс эквивалентности из $D_{\le}(n)$, получаемый из $n_1 + n_2 + \ldots + n_k$ всеми перестановками этих слагаемых, так что существует биекция между классами эквивалентности на множестве элементов k-уровня из $D_{\le}(n)$ и множеством элементов k-уровня из $D(n)$, $0 \le k \le n$.

Существует монотонная сюръекция $sur(D_{\le}(n)) = D(n)$.

Решетка Юнга J имеет следующую связь с $D(n)$: $|D(n)| =$ |множество элементов уровня n решетки J|.

Имеет место следующая связь между $D_{\le}(n)$ композиций числа n и ПР-кодами над алфавитом $A = \{0, \ldots, n\}$: мощность множества слов длины $n + k$ и веса $\omega \le n$, $0 < \omega$, в $L_{ПРК}(m, n+k)$ полностью определяется всеми подходящими расстановками нулей в элементах из $D_{\le}(i)$, $1 \le i \le n$. Поэтому существует сюръекция $n+k$-уровня и веса $\omega \le n$ решетки $L_{ПРК}(m, n+k)$ на $D_{\le}(n)$. ■

В свете вышесказанного становится очевидным, что теория L-кодирования не ограничена рамками проблематики лишь равномерных кодов, потому что, например, элементы любой решетки $D_{\le}(n)$ композиций, как и доминирующего порядка $D(n)$, могут быть взяты и как слова неравной длины, а потому и как неравномерный код постоянного веса (см. рис. 2).

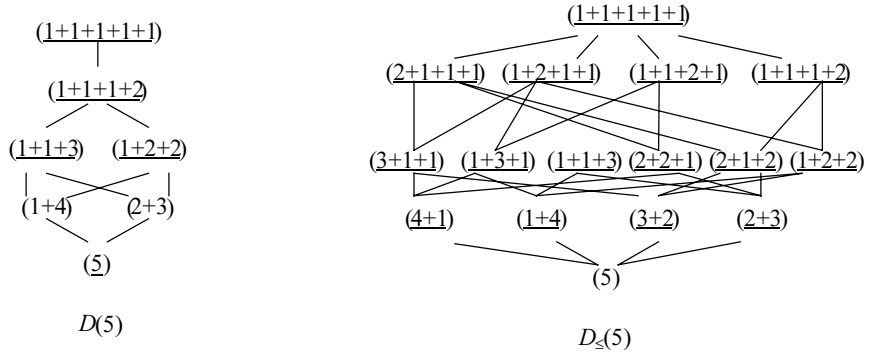


Рис. 2

К тому же изоморфизм $D_{\leq}(n) \cong Bool(n–1)$ можно использовать для кодирования с переходом от слов 11…1, 21…1, …, 1…12,…, *n*–11, *n* соответственно длины от *n* до 1 над алфавитом {1, …, *n*} к словам равной длины *n*–1 над алфавитом {0, 1}, а также для обратного перехода. Можно также использовать вложения решетки $D_{\leq}(n)$ в подходящую решетку $D_{\leq}(n + k)$, $k = 1, \ldots$, при расширении кода и обратно: использовать наложения $D_{\leq}(n + k)$ на $D_{\leq}(n)$ при сжатии (сужении) кода, выкалывании и т.п.; эти же приемы, естественно, можно использовать для $L_{\text{ПРК}}(m, n)$ при вложениях ее в $L_{\text{ПРК}}(m+k, n)$ или $L_{\text{ПРК}}(m+k, n+t)$, $k, t \geq 1$, или при их наложениях на $L_{\text{ПРК}}(m, n)$. Тем самым получаем различные способы и кодирования, и шифрования (точнее, кодошифрования – шифрования закодированного сообщения).

3. Структуры матроидного типа: матроиды и *c*-матроиды

Множество *S* вместе с оператором замыкания $A \to \bar{A}$ называется матроидом [10, 11], если для всех $A \subseteq S$ и любых $p, q \in S$ выполняются аксиомы замены Штейница и конечного базиса.

Как известно [10, гл. II, § 3, теорема 2.29 (Биркгоф – Уитни)], множество точек $T = \{p: p \in L_\gamma\}$ геометрической решетки L_γ вместе с оператором замыкания $\varphi: T' \to \bar{T}' = \{p: p \leq \sup(T'),\ p \in T, \varnothing \neq T' \subseteq T\}$ и аксиомой замены Штейница является простым матроидом.

В частности, из этого предложения и теоремы 1 извлекаем следующее утверждение.

Утверждение 2. Множество *T* точек булевой решетки двоичного ПР-кода над алфавитом $A = \{0, 1\}$ с оператором замыкания φ и аксиомой замены Штейница является простым матроидом. Если к *T* добавить о и взять пару: множество $T \cup \{o\}$ вместе с отношением порядка ≤, являющимся ограничением порядка булевой решетки на $T \cup \{o\}$, то пара $<T \cup \{o\}, \leq>$ превращается в суперматроид (определение суперматроида и его свойства см. в [12, 13]). ■

Определение [2]. Множество всех *c*-элементов $C \subseteq L$ вместе с оператором замыкания φ называется *c*-матроидом, если для всех $C' \subseteq C$, $C' \neq \varnothing$, и любых $c_1, c_2 \in C$ выполняется следующая аксиома замены:

из $c_1 \notin \varphi(C')$ и $c_1 \in \varphi(C' \cup c_2)$ следует существование такого c_2', что $o < c_2' \leq c_2$ и $c_2' \in \varphi(C' \cup c_1)$.

Как указано в теореме 1, множество всех *c*-элементов решетки L_A является *c*-матроидом, а при добавлении к ним всех элементов рангов, соответствующих рангам *c*-элементов, оказывается суперматроидом.

Исследование *c*-матроидов связано также с 25-й проблемой Скорнякова [14]: исследовать дедекиндовы структуры, в которых каждый элемент представляется в форме $a = \sum_{i=1}^{n(a)} La_i$, где La_i – цепи, в частности конечные (дедекиндовы структуры – это модулярные решетки), см. [2].

Заключение. Структуры других моделей

Приведем здесь кратко факты, подтверждающие, что и многие другие модели, используемые в кодировании и криптографии, имеют структуры, являющиеся *s*-множествами, в частности решетками.

Геометрические коды, как известно, строятся над геометриями. С геометриями, в которых две различные точки определяют единственную прямую (однозначные геометрии), выпуклыми геометриями, матроидами и антиматроидами ассоциированы геометрические и слабо полудистрибутивные решетки [15]. По этому поводу отметим следующее. Неоднозначные геометрии (две различные точки не обязательно определяют единственную прямую, например, как в геометрии Мебиуса) ассоциированы не с решетками, а с геометрическими *ML*-решетками и *p*-матроидами [2, 3].

О структурах моделей дискретных систем, в том числе автоматов и ассоциированных с ними языков, см. [16 – 18], где показано, что эти структуры являются решетками или полурешетками.

О семиотических моделях информации, моделях неопределенности и соответствующих им *s*-множествах и решетках см. [7, 19, 20].

Проблемы и вопросы теории кодирования и криптографии приведены в [8, 9], см. также [1, 3 – 5, 10].

ЛИТЕРАТУРА

1. *Кутьин А.М.* Коды и решетки // Вестник ТГУ. Приложение. 2006. № 17. С. 30 – 34.
2. *Кутьин А.М.* Р-матроиды // Дискретная математика. 2005. Т. 17. Вып. 3. С. 146 – 160.
3. *Кутьин А.М.* Проблемы теории кодирования и теория геометрий // Проблемы информатизации региона. Красноярск: Изд-во КТГУ, 2001. С. 90 – 103.
4. *Кутьин А.М.* Открытые вопросы теории кодирования и криптографии. I // Проблемы информатизации региона. Красноярск: Изд-во КТГУ, 2001. С. 74 – 91.
5. *Кутьин А.М.* Информация, безопасность, кодирование и решетки. III // Проблемы информатизации региона: Шестая Всерос. науч.-практич. конф.: Доклады. Красноярск: Изд-во КТГУ, 2000. С. 74 – 79.
6. *Кутьин А.М.* Вопросы теории кодирования и решетки // Безопасность информационных технологий. М.: МИФИБ, 2001. № 4. С. 79 – 84.
7. *Кутьин А.М.* О моделях информации. II // Информационная реальность и цивилизация. Красноярск: Изд-во САА, 1998. С. 73 – 81.
8. *Конвей Дж.*, *Слоэн Н.* Упаковки шаров, решетки и группы. Т. 1. М.: Мир, 1990.
9. *Мак-Вильямс Ф.Дж.*, *Слоэн Н.Дж.А*. Теория кодов, исправляющих ошибки. М.: Связь, 1979.
10. *Айгнер М.* Комбинаторная теория. М.: Мир, 1982. 558 с.
11. *Биркгоф Г.* Теория решеток. М.: Мир, 1984.
12. *Емеличев В.А*., *Ковалев М.М*., *Кравцов М.К*. Многогранники, графы, оптимизация. М.: Наука, 1981.
13. *Ковалев М.М.* Матроиды в дискретной оптимизации. Минск: Изд-во Университетское, 1987.
14. *Скорняков Л.А*. Дедекиндовы структуры с дополнениями и регулярные кольца. М.: Физматгиз, 1961.
15. *Горбунов В.А.* Алгебраическая теория квазимногообразий. Новосибирск: Научная книга, 1999.
16. *Агибалов Г.П.* Дискретные автоматы на полурешетках. Томск: Изд-во Том. ун-та, 1993.
17. *Богомолов А.М.*, *Салий В.Н.* Алгебраические основы теории дискретных систем. М.: Наука, 1997.
18. *Салий В.Н*. Представление языков в решеточных автоматах // Методы и системы техн. диагностики. 1992. № 17. С. 33 – 36.
19. *Кутьин А.М.* К теории неопределенности: модели и структуры // Вестник КГТУ. Математические методы и моделирование. 2003. № 30. С. 14 – 27.
20. *Кутьин А.М.* Теория направленных *s*-множеств // Вестник КГТУ. 2001. Вып. 26. С. 117 – 126.

2008 Теоретические основы прикладной дискретной математики № 1(1)

DOI 10.17223/20710410/1/4

УДК 512.816+512.548.7

ТРЕХБАЗИСНЫЕ КВАЗИГРУППЫ С ОБОБЩЕННЫМ ТОЖДЕСТВОМ УОРДА

В.А. Кыров

Горно-Алтайский государственный университет

E-mail: kfizika@gasu.ru

В данной работе изучается трехбазисная частичная квазигруппа с обобщенным тождеством Уорда. Доказывается, что эта квазигруппа является феноменологически симметричной. Изучаются феноменологически симметричные подквазигруппы.

Ключевые слова: *квазигруппа*, *тождество Уорда*, *трехбазисная квазигруппа с обобщенным тождеством Уорда*, *трехбазисная феноменологически симметричная квазигруппа.*

Как известно, квазигруппой Уорда называется квазигруппа $(C,\circ)$ с тождеством Уорда [1]:

$$(a\circ b)\circ(c\circ b) = (a\circ c),$$

где $\circ$ – бинарная операция; a, b, c – произвольные элементы множества C. В теории квазигрупп доказывается, что любая квазигруппа Уорда изотопна некоторой группе. В данной работе изучаются трехбазисные частичные квазигруппы с тождеством, обобщающим тождество Уорда, и устанавливается связь таких квазигрупп с физическими структурами. Квазигрупповая терминология приводится по монографии [2].

1. Трехбазисная квазигруппа с обобщенным тождеством Уорда

Пусть B – некоторое множество, а $\Omega_B{}^n \subset B^n$ – некоторое подмножество.

Определение 1. *Трехбазисной квазигруппой с обобщенным тождеством Уорда* называется частичная трехбазисная квазигруппа $(B, \Omega_B{}^n, B, \varphi)$ с отображением φ: $B\times\Omega_B{}^n \to B$ и системой аксиом К1 – К3.

К1. $\forall\ \langle x_1, \ldots, x_n\rangle, \langle b_1, \ldots, b_n\rangle \in \Omega_B{}^n\ \exists!\ y \in \Omega_B{}^n$: $\varphi(x_k, y) = b_k$, $k = 1, \ldots, n$.

К2. $\forall\ y \in \Omega_B{}^n\ \forall\ b \in \Omega_B\ \exists!\ x \in B$: $\varphi(x, y) = b$, где $\Omega_B \subset B$.

Построим два вспомогательных отображения: F_z: $\Omega_B{}^n \to \Omega_B{}^n$, $z = \langle z_1, \ldots, z_n\rangle \in \Omega_B{}^n$: $F_z(y) = (\varphi(z_1 y), \ldots, \varphi(z_n y))$, $y \in \Omega_B{}^n$, и F_w: $B\to B$, $w \in \Omega_B$: $F_w(x) = \varphi(xw)$, $x \in B$. Из аксиом К1 и К2 следует биективность построенных отображений.

К3. Обобщенное тождество Уорда:

$$\varphi\,(\varphi(x, a_1, \ldots, a_n), \varphi(y_1, a_1, \ldots, a_n), \ldots, \varphi(y_n, a_1, \ldots, a_n)) = \varphi(x, y_1, \ldots, y_n), \qquad (1)$$

где $\langle y_1, \ldots, y_n\rangle, \langle a_1, \ldots, a_n\rangle \in \Omega_B{}^n$, $x \in B$.

По трехбазисной квазигруппе с обобщенным тождеством Уорда $(B, \Omega_B{}^n, B, \varphi)$ можно построить квазигруппу Уорда $(\Omega_B{}^n, \rho)$ с бинарной операцией

$$\rho(X, Y) = (\varphi(x_1, y_1, \ldots, y_n), \ldots, \varphi(x_n, y_1, \ldots, y_n)),$$

где $X = \langle x_1, \ldots, x_n\rangle$, $Y = \langle y_1, \ldots, y_n\rangle \in \Omega_B{}^n$. Аксиомы квазигруппы $(\Omega_B{}^n, \rho)$ следуют из К1 – К3, причем тождество Уорда имеет вид

$$\rho(\rho(X, A), \rho(Y, A)) = \rho(X, Y). \qquad (2)$$

Теорема 1. Квазигруппа Уорда $(\Omega_B{}^n, \rho)$ изотопна группе $(\Omega_B{}^n, \cdot)$ с бинарной операцией $X\cdot Y = \rho(X, \theta(Y))$, где $\theta(A) = \rho(E, A)$, $\rho(A, A) = E$.

Доказательство. Сначала в тождестве (2) положим $Y = A$, тогда $\rho(\rho(X, A), \rho(A, A)) = \rho(X, A)$. Так как точки A и X произвольные, то по свойству однозначной разрешимости уравнений квазигруппы $\rho(A, A) = E$, следовательно, $\rho(A, E) = A$. Обозначим $\theta(A) = \rho(E, A)$ (из определения квазигруппы следует биективность этого отображения). Несложно также доказать, что $\rho(E, \rho(X, A)) = \rho(A, X)$, $\theta(\theta(X)) = X$. Рассмотрим теперь квазигруппу, изотопную квазигруппе Уорда с бинарной операцией $X\cdot Y = \rho(X, \theta(Y))$. Однозначная разрешимость для новой операции очевидна.

Лемма. Квазигруппа $(\Omega_B{}^n, \cdot)$ является лупой с единицей E.

Действительно, $X\cdot E = \rho(X, \theta(E)) = \langle\theta(E) = E\rangle = X$; $E\cdot Y = \rho(E, \theta(Y)) = \theta(\theta(X)) = X$. ■

Докажем ассоциативность. Рассмотрим тождественное равенство

$$(X\cdot A)\cdot\theta(Y\cdot A) = X\cdot\theta(Y), \qquad (*)$$

которое вытекает из (2). Очевидно, $X\cdot Y = \rho(X, \theta(Y)) = \rho(E, \rho(\theta(Y), X)) = \rho(E, \theta(Y)\cdot\theta(X)) = \theta(\theta(Y)\cdot\theta(X))$. Полагая в тождестве (*) $Y = E$ и пользуясь выше доказанной леммой, имеем $(X\cdot A)\cdot\theta(A) = X$. Тогда в тождестве (*) берем $A = Z$, $Y\cdot A = \theta(U)$. Заметим, что $(Y\cdot A)\cdot\theta(A) = \theta(U)\cdot\theta(A) = Y$. Элементарные расчеты дают ассоциативность $(X\cdot Z)\cdot U = X\cdot(Z\cdot U)$. ■

Докажем некоторые свойства трехбазисной квазигруппы с обобщенным тождеством Урда.

Теорема 2. Справедливы следующие свойства трехбазисной квазигруппы $(B, \Omega_B^n, B, \varphi)$ с обобщенным тождеством Урда (1):

1) $\varphi(a_1, a_1, \ldots, a_n) = e_1, \ldots, \varphi(a_n, a_1, \ldots, a_n) = e_n, \forall A = <a_1, \ldots, a_n> \in \Omega_B^n, E = <e_1, \ldots, e_n> \in \Omega_B^n$ – постоянная точка;

2) $\varphi(x, e_1, \ldots, e_n) = x$;

3) $\varphi(e_1, a_1, \ldots, a_n) = \theta_1(a_1, \ldots, a_n), \ldots, \varphi(e_n, a_1, \ldots, a_n) = \theta_n(a_1, \ldots, a_n)$;

4) $\varphi(e_1, \theta_1(a_1, \ldots, a_n), \ldots, \theta_n(a_1, \ldots, a_n)) = a_1, \ldots, \varphi(e_n, \theta_1(a_1, \ldots, a_n), \ldots, \theta_n(a_1, \ldots, a_n)) = a_n$.

Доказательство. Воспользуемся связью трехбазисной квазигруппы $(B, \Omega_B^n, B, \varphi)$ с квазигруппой Урда (Ω_B^n, ρ). Тогда свойства 1 – 4 будут результатом покомпонентной записи соответствующих свойств квазигруппы (Ω_B^n, ρ). Другими словами, 1 – это покомпонентная запись равенства $\rho(A, A) = E$, 2 – соответственно тождества $\rho(A, E) = A$, 3 – равенства $\rho(E, A) = \theta(A)$ и, наконец, 4 – тождества $\theta(\theta(X)) = X$. ■

2. Трехбазисная феноменологически симметричная квазигруппа ранга $(n+1, 2)$ и ее связь с трехбазисной квазигруппой с обобщенным тождеством Урда

Рассмотрим три множества M, N, B и отображение f: $M \times N \to B$, называемое метрическим. Пусть выполняются аксиомы [2]:

А1. $\forall <i_1, \ldots, i_n> \in \Omega_M^n \ \forall <b_1, \ldots, b_n> \in \Omega_B^n \ \exists ! \ \alpha \in N: f(i_k\alpha) = b_k$, $k = 1, \ldots, n$, где $\Omega_M^n \subset M^n$ и $\Omega_B^n \subset B^n$.

А2. $\forall \ \alpha \in \Omega_N \ \forall \ b \in B \exists ! \ i \in M: f(i\alpha) = b$, где $\Omega_N \subset N$.

А3. $\forall <i_0, i_1, \ldots, i_n> \in M \times \Omega_M^n \ \forall <\alpha_0, \alpha_1> \in N \times \Omega_N$ существует связь

$$f_{00} = g(f_{01}, f_{11}, \ldots, f_{n1}, f_{10}, \ldots, f_{n0}),$$

где $f_{mn} = f(i_m, \alpha_n)$.

Построим два вспомогательных отображения: F_j: $N \to \Omega_B^n$, $j = <j_1, \ldots, j_n> \in \Omega_M^n$, с явным видом $F_j(\alpha) = (f(j_1\alpha), \ldots, f(j_n\alpha))$, и F_γ: $M \to B$, $\gamma \in \Omega_N$: $F_\gamma(i) = f(i\gamma)$. Из аксиом А1 и А2 следует биективность этих отображений.

Определение 2. Говорят, что на множествах M, N, B с метрическим отображением f: $M \times N \to B$ задана *физическая структура ранга* $(n+1, 2)$, если выполняются аксиомы А1 – А3.

Из аксиом А1, А2 также следует, что на множествах M, N, B определена частичная трехбазисная квазигруппа с операцией f. По аксиоме А3 это специальная трехбазисная квазигруппа, которую будем называть *феноменологически симметричной ранга* $(n+1, 2)$ и обозначать (M, N, B, f).

Обозначим $x = f(i\gamma)$, $y_1 = f(j_1\alpha)$, …, $y_n = f(j_n\alpha)$. Тогда метрическое отображение можно переписать так:

$$f(i\alpha) = f(F^{-1}{}_\gamma(x), F^{-1}{}_j(y_1, \ldots, y_n)) = \overline{f}\,(x, y_1, \ldots, y_n).$$

Построенное отображение $\overline{f}: B \times \Omega_B^n \to B$, очевидно, удовлетворяет таким аксиомам:

Б1. $\forall <x_1, \ldots, x_n>, <b_1, \ldots, b_n> \in \Omega_B^n \ \exists ! \ y \in B$: $\overline{f}\,(x_k y) = b_k$, $k = 1, \ldots, n$.

Б2. $\forall \ y \in \Omega_B^n \ \forall \ b \in \Omega_B \exists ! \ x \in B$: $\overline{f}\,(x, y) = b$.

Б3. $\forall <x_0, x_1, \ldots, x_n> \in B \times \Omega_B^n \ \forall <z_0, z_1> \in \Omega_B^n \times \Omega_B^n$ существует связь

$$\overline{f}_{00} = g(\overline{f}_{01}, \overline{f}_{11}, \ldots, \overline{f}_{n1}, \overline{f}_{10}, \ldots, \overline{f}_{n0}),$$

где $\overline{f}_{ij} = \overline{f}\,(x_i, z_j)$.

Таким образом, отображение $\overline{f}$ задает физическую структуру ранга $(n+1, 2)$. Другими словами, определена феноменологически симметричная трехбазисная квазигруппа ранга $(n+1, 2)$, которую обозначим $(B, \Omega_B^n, B, \overline{f}\,)$.

По квазигруппе $(B, \Omega_B^n, B, \overline{f}\,)$ строится новая частичная трехбазисная квазигруппа $(B^n, \Omega_B^n, B^n, \overline{F}\,)$ с бинарной операцией

$$\overline{F}\,(XY) = (\overline{f}\,(x_1, y_1, \ldots, y_n), \ldots, \overline{f}\,(x_n, y_1, \ldots, y_n)), \tag{3}$$

где $X = (x_1, \ldots, x_n) \in B^n$; $Y = (y_1, \ldots, y_n) \in \Omega_B^n$.

Теорема 3. Для трехбазисной квазигруппы $(B^n, \Omega_B^n, B^n, \overline{F}\,)$ выполняется тождество, обобщающее ассоциативность

$$X(YZ) = (XY)Z,$$

где $X \in B^n$, $Y, Z \in \Omega_B^n$.

Доказательство можно найти в работе [3].

Если в бинарной операции (3) $X \in \Omega_B^n$, то $\overline{F}\,(XY) \in \Omega_B^n$. Из теоремы 3 тогда следует, что Ω_B^n является группой с бинарной операцией (3) и единицей $E = (e_1, \ldots, e_n) = (f(j_1\gamma), \ldots, f(j_n\gamma))$. Поэтому трехбазисная ква-

зигруппа (B^n, $\Omega_B{}^n$, B^n, $\overline{F}$) является почти просто транзитивной группой преобразований множества B^n с параметрической группой $\Omega_B{}^n$. Тогда, очевидно, феноменологически симметричная трехбазисная квазигруппа ранга (n+1, 2) (B, $\Omega_B{}^n$, B, $\overline{f}$) является транзитивной группой преобразований множества B с параметрической группой $\Omega_B{}^n$.

Теорема 4. Тождество из аксиомы Б3 для феноменологически симметричной трехбазисной квазигруппы ранга (n+1, 2) (B, $\Omega_B{}^n$, B, $\overline{f}$) эквивалентно обобщенному тождеству Уорда:

$$\tilde{f}\,(\,\tilde{f}\,(x, a_1, \ldots, a_n), \tilde{f}\,(y_1, a_1, \ldots, a_n), \ldots, \tilde{f}\,(y_n, a_1, \ldots, a_n)) = \tilde{f}\,(x, y_1, \ldots, y_n) \tag{4}$$

или

$$\tilde{f}\,(\,\tilde{f}\,(x, a_1, \ldots, a_n), \tilde{f}\,(y_1, a_1, \ldots, a_n), \ldots, \tilde{f}\,(y_n, a_1, \ldots, a_n)) =$$

$$= \tilde{f}\,(\,\tilde{f}\,(x, b_1, \ldots, b_n), \tilde{f}\,(y_1,b_1,\ldots,b_n),\ldots, \tilde{f}\,(y_n,b_1,\ldots,b_n)\,), \tag{4′}$$

где $\tilde{f}\,(x, y_1, \ldots, y_n) = \overline{f}\,(x, (y_1, \ldots, y_n)^{-1})$, $\forall < y_1, \ldots, y_n>, < a_1, \ldots, a_n>, < b_1, \ldots, b_n> \in \Omega_B{}^n$, $x \in B$.

Доказательство. Из определения отображений $\overline{f}$ и $\tilde{f}$ следует выполнимость аксиомы Б2, т. е. однозначная разрешимость этих функций относительно первого аргумента. Поэтому, разрешая тождество (4′) относительно первого аргумента левой части, получаем аксиому Б3.

Докажем теперь обратное. Для этого в тождестве теоремы 3 положим $Y = A$, $Z = (YA)^{-1}$. После приведения подобных имеем $(XA)(YA)^{-1} = XY^{-1}$. Каждая компонента этого тождества совпадает с (4). ■

Из теоремы 4 следует, что четверка (B, $\Omega_B{}^n$, B, $\tilde{f}$) также образует феноменологически симметричную трехбазисную квазигруппу ранга (n+1, 2), изотопную квазигруппе (B, $\Omega_B{}^n$, B, $\overline{f}$).

Из теоремы 4 вытекает, что трехбазисная феноменологически симметричная квазигруппа (B, $\Omega_B{}^n$, B, $\tilde{f}$) ранга (n+1, 2) является трехбазисной квазигруппой с обобщенным тождеством Уорда (1). Верно и обратное.

Теорема 5. Трехбазисная квазигруппа (B, $\Omega_B{}^n$, B, φ) с обобщенным тождеством Уорда является трехбазисной феноменологически симметричной квазигруппой ранга (n+1, 2).

Доказательство. Пусть (B, $\Omega_B{}^n$, B, φ) – трехбазисная квазигруппа с обобщенным тождеством Уорда (1). С ее помощью строится квазигруппа Уорда ($\Omega_B{}^n$, ρ), изотопная группе. Для этого изотопа, очевидно, можно записать тождество (4). Тогда все свойства феноменологически симметричной трехбазисной квазигруппы ранга (n+1, 2) будут выполнены. ■

Таким образом, существует два способа определения физической структуры ранга (n+1, 2). Первый основан на аксиоме А3, а второй – на обобщенном тождестве Уорда (1).

3. Феноменологически симметричные трехбазисные подквазигруппы ранга (k+1, 2)

Рассмотрим феноменологически симметричную трехбазисную квазигруппу (B, $\Omega_B{}^n$, B, $\overline{f}$) ранга (n+1, 2), которая, напомним, является группой преобразований множества B. Из аксиомы Б1 вытекает транзитивность этой группы преобразований. Рассмотрим ее стационарную подгруппу $G_{e_n} = (B', \Omega'_B{}^{n-1}, B', \overline{f}\,)$ [4], оставляющую на месте точку e_n, $B' \subset B$ – совокупность точек, подвижных относительно этой подгруппы. Заметим, что параметрическая группа $\Omega'_B{}^{n-1}$ стационарной подгруппы G_{e_n} изоморфна группе, состоящей из совокупности точек $X = <x_1, \ldots, x_{n-1}, e_n>$, $Y = <y_1, \ldots, y_{n-1}, e_n> \in \Omega'_B{}^{n-1} \subset \Omega_B{}^n$: $\overline{F}\,(XY) \in \Omega'_B{}^{n-1}$. Тогда

$$\overline{f}\,(x, y_1, \ldots, y_{n-1}, e_n) = x',\ \overline{f}\,(e_p, y_1, \ldots, y_{n-1}, e_n) = y_p,\ \overline{f}\,(e_n, y_1, \ldots, y_{n-1}, e_n) = e_n, \tag{5}$$

где $p = 1,\ldots, n-1$. Несложно проверить, что для множества (B', $\Omega'_B{}^{n-1}$, B', $\overline{f}$) выполняются аксиомы Б1 – Б3. Легко доказываются аксиомы Б1, Б2. Для проверки аксиомы Б3 необходимо в тождестве (4) положить $a_n = y_n = e_n$, а затем воспользоваться равенствами (5). Таким образом, (B', $\Omega'_B{}^{n-1}$, B', $\overline{f}$) – феноменологически симметричная частичная трехбазисная подквазигруппа ранга (n, 2) в (B, $\Omega_B{}^n$, B, $\overline{f}$). Эта подквазигруппа задает физическую структуру ранга (n, 2).

Две феноменологически симметричные квазигруппы (B, $\Omega_B{}^n$, B, $\overline{f}$) и (B', $\Omega'_B{}^n$, B', $\overline{f}'$) ранга (n+1, 2) будем называть эквивалентными, если группы $\Omega_B{}^n$ и $\Omega'_B{}^n$ изоморфны.

Теорема 6. Две феноменологически симметричные трехбазисные подквазигруппы (B', $\Omega'_B{}^{n-1}$, B', $\overline{f}'$) и (B'', $\Omega''_B{}^{n-1}$, B'', $\overline{f}''$) ранга (n, 2) квазигруппы (B, $\Omega_B{}^n$, B, $\overline{f}$) с неподвижными точками e_n и e'_n, эквивалентны.

Доказательство. Из определения стационарной подгруппы следует, что группы (B', $\Omega'_B{}^{n-1}$, B', $\overline{f}'$) и (B'', $\Omega''_B{}^{n-1}$, B'', $\overline{f}''$) сопряжены [4], следовательно, изоморфны. ■

Если в стационарной подгруппе $G_{e_n} = (B', \Omega'_B{}^{n-1}, B', \overline{f}\,)$ зафиксировать еще одну точку, то, очевидно, получится феноменологически симметричная частичная трехбазисная подквазигруппа ранга $(n, 2)$ в $(B, \Omega_B{}^n, B, \overline{f}\,)$, значит, по теореме 6 эта подквазигруппа будет единственна, с точностью до эквивалентности.

Продолжая фиксировать точки, получим феноменологически симметричную частичную трехбазисную подквазигруппу ранга $(k+1, 2)$ в $(B, \Omega_B{}^n, B, \overline{f}\,)$, $k < n$, причем имеет место следующая теорема, аналогичная 6.

Теорема 6′. Две феноменологически симметричные трехбазисные подквазигруппы $(B', \Omega'_B{}^k, B', \overline{f}'\,)$ и $(B'', \Omega''_B{}^k, B'', \overline{f}''\,)$ ранга $(k+1, 2)$ квазигруппы $(B, \Omega_B{}^n, B, \overline{f}\,)$ с неподвижными точками $\{e_m\}$ *и* $\{e'_m\}$, эквивалентны. ■

4. Пример

Рассмотрим феноменологически симметричную трехбазисную квазигруппу ранга (4, 2) на проективной прямой RP. В таком случае $B = \mathrm{RP}$, а метрическое отображение – это проективное преобразование в RP:

$$\overline{f} = \frac{ax+b}{cx+d},\ u = \frac{ae_1+b}{ce_1+d},\ v = \frac{ae_2+b}{ce_2+d},\ w = \frac{ae_3+b}{ce_3+d},$$

где $<e_1, e_2, e_3> \in \Omega_{\mathrm{RP}}$, матрица из коэффициентов обратима. Обобщенное тождество Уорда (4) выражается через сложное отношение четырех точек. Рассмотрим теперь феноменологически симметричную трехбазисную подквазигруппу ранга (3, 2), т. е. зафиксируем точку e_3:

$$\overline{f} = \frac{ax+b}{cx+d},\ u = \frac{ae_1+b}{ce_1+d},\ v = \frac{ae_2+b}{ce_2+d},\ e_3 = \frac{ae_3+b}{ce_3+d}.$$

Эта подквазигруппа эквивалентна полной аффинной группе преобразований прямой R (теорема 6):

$$\overline{f} = ax + b.$$

И наконец, рассмотрим феноменологически симметричную трехбазисную подквазигруппу ранга (2, 2), т.е. зафиксируем две точки e_2 и e_3:

$$\overline{f} = \frac{ax+b}{cx+d},\ u = \frac{ae_1+b}{ce_1+d},\ e_2 = \frac{ae_2+b}{ce_2+d},\ e_3 = \frac{ae_3+b}{ce_3+d}.$$

В явном виде метрическое отображение следующее:

$$\overline{f} = \frac{xu(e_2+e_3-e_1) - xe_2e_3 - ue_2e_3 + e_1e_2e_3}{xu - xe_1 - ue_1 - e_2e_3 + e_1(e_2+e_3)}.$$

Оно задает групповую бинарную операцию в $B' = \mathrm{RP}/\{e_2, e_3\}$ с единицей e_1, которая изоморфна мультипликативной группе поля действительных чисел.

ЛИТЕРАТУРА

1. *Chatterjea S.K.* On Ward quasigroups // Pure Math. Manuscript. 1987. V. 6. P. 31 – 34.
2. *Белоусов В.Д.* Основы теории квазигрупп и луп. М.: Наука, 1967.
3. *Симонов А.А.* Обобщенное матричное умножение как эквивалентное представление теории физических структур // Приложение к книге Ю.И. Кулакова «Теория физических структур». М., 2004.
4. *Горбацевич В.В.*, *Онищик А.Л.* Группы Ли преобразований // Итоги науки и техники. Современные проблемы математики. Фундаментальные направления. 1988. Т. 20. С. 108 – 248.

2008 Теоретические основы прикладной дискретной математики № 1(1)

DOI 10.17223/20710410/1/5

УДК 519.7

ЛИНЕЙНЫЕ СТРУКТУРЫ ГРУПП ПОДСТАНОВОК НАД КОНЕЧНЫМ МОДУЛЕМ[1]

М.А. Пудовкина

Московский инженерно-физический институт (*государственный университет*)

E-mail: maricap@rambler.ru

Описание отображений с линейными структурами важно для синтеза и анализа шифрсистем, поскольку нетривиальная линейная структура является слабостью криптографического отображения. В частности, позволяет применить дифференциальный метод криптоанализа или метод гомоморфизмов. В данной работе на групповом языке описаны все биективные отображения над конечным модулем с линейной структурой.

Ключевые слова: *линейная структура*, *модуль*, *группа подстановок*, *импримитивная группа*, *сплетение групп.*

Линейная структура отображений (блочных шифрсистем) рассматривалась в работах [1 – 4] и др. Некоторые свойства отображений с линейными структурами приведены, например, в работе [5]. Наличие нетривиальной линейной структуры является слабостью криптографического отображения, в частности позволяет применять метод дифференциального криптоанализа или строить фактор-системы для данной криптографической системы, используя метод гомоморфизмов в явном виде [2] . Таким образом, описание отображений, обладающих линейными структурами, важно для синтеза и анализа шифрсистем.

В данной работе на групповом языке описаны все биективные отображения над конечным модулем с линейной структурой.

Будем придерживаться следующих обозначений: N_0 – множество натуральных чисел с нулем, $m \in N_0$, $m \geq 2$; R – конечное ассоциативное кольцо с единицей; p – простое число; $H \in \{GF(p^m), V_m(p), Z_{p^m}, R\text{-модуль}\}$; $\overline{a,b} = a, a+1, \ldots, b$, $a < b$; $S(X)$ – симметрическая группа подстановок на множестве X; $\pi, \sigma \in S(H)$; 0 – нулевой элемент H; $d(\pi, \sigma) = |\{\alpha \in H | \alpha^\pi \neq \alpha^\sigma\}|$; $<\alpha>$ – группа, порожденная элементом $\alpha \in H \backslash 0$. Отметим, что поле $GF(p^m)$ рассматривается как m-мерное векторное пространство над $GF(p)$.

Напомним (см., например, [5]), что отображение $\pi \in S(H)$ обладает линейной структурой, если в H существует ненулевой элемент α, такой, что $(\beta+\alpha)^\pi = \beta^\pi + \gamma_\alpha$ для любого $\beta \in H$, где $\gamma_\alpha \in H \backslash 0$ – некоторый фиксированный элемент. Элемент α называется линейным транслятором отображения π.

Пусть

$$\Pi_{\alpha,\gamma} = \left\{\pi \in S(H) \,|\, (\beta+\alpha)^\pi - \beta^\pi = \gamma,\ \forall \beta \in H\right\},\ \alpha, \gamma \in H \backslash 0,$$

$$\Pi_{W,W} = \left\{\pi \in S(H) \,|\, (\beta+\alpha)^\pi - \beta^\pi \in W,\ \forall \beta \in F, \forall \alpha \in W\right\},$$

где W – произвольный нетривиальный подмодуль R-модуля H. В частности, $\Pi_{W,W} = \Pi_{\alpha,\alpha}$, где $W = <\alpha>$. Существование у отображения подмодуля W, для которого $\Pi_{W,W} \neq e$, можно рассматривать как обобщение понятия линейной структуры. Его наличие может привести, например, к применению метода гомоморфизмов.

Для $H \in \{GF(2^m), V_m(2)\}$ множества $\Pi_{\alpha,\gamma}$, $\Pi_{W,W}$ описаны автором в работе [4].

1. Описание множества $\Pi_{W,W}$

Для произвольного подмодуля W R-модуля H приведем характеризацию множества $\Pi_{W,W}$ на групповом языке.

Теорема 1. Пусть W – произвольный нетривиальный подмодуль R-модуля H, $|H| = b$, $|W| = d$. Тогда множество $\Pi_{W,W}$ является импримитивной группой из $S(H)$ с системой импримитивности $\{\beta+W | \beta \in H\}$. Группа $\Pi_{W,W}$ подобна группе $S_d \int S_{d^{-1}b}$.

Доказательство. Покажем, что $\Pi_{W,W}$ является группой. Если $\pi_1, \pi_2 \in \Pi_{W,W}$, то для любого $\alpha \in W$ из равенств

$$(\beta+\alpha)^{\pi_1\pi_2} - \beta^{\pi_1\pi_2} = (\beta^{\pi_1}+\alpha)^{\pi_2} - (\beta^{\pi_1})^{\pi_2} = \beta^{\pi_1\pi_2} + \alpha' - \beta^{\pi_1\pi_2} = \alpha' \in W$$

[1] Работа выполнена при поддержке гранта Президента РФ НШ №4.2008.10.

следует, что $\pi_1\pi_2 \in \Pi_{W,W}$. Очевидно, что $e \in \Pi_{W,W}$. Осталось доказать, что $\pi^{-1} \in \Pi_{W,W}$ для любой $\pi \in \Pi_{W,W}$. Если это не так, то существуют $\gamma \in H$, $\alpha \in W$ и подстановка $\pi \in \Pi_{W,W}$ такие, что $(\gamma+\alpha)^{\pi^{-1}} - \gamma^{\pi^{-1}} \notin W$. Положим $\beta = \gamma^{\pi^{-1}}$. Тогда из равенства

$$(\gamma+\alpha)^{\pi^{-1}} - \gamma^{\pi^{-1}} = (\beta^{\pi}+\alpha)^{\pi^{-1}} - (\beta^{\pi})^{\pi^{-1}} = (\beta+\alpha')^{\pi\pi^{-1}} - (\beta^{\pi})^{\pi^{-1}} = \alpha' \in W$$

получаем противоречие с предположением, что $(\gamma+\alpha)^{\pi^{-1}} - \gamma^{\pi^{-1}} \notin W$. Значит, $\Pi_{W,W}$ – группа.

Для любой подстановки $\pi \in \Pi_{W,W}$ равенство $(\beta+W)^{\pi} = \beta^{\pi} + W$ эквивалентно тому, что π: $\beta+W \to \beta^{\pi}+W$ для каждого $\beta \in H$. Группа, порождаемая всеми такими подстановками, является импримитивной с системой импримитивности $\{\beta+W | \beta \in H\}$ и совпадает с группой $S_d \int S_{d^{-1}b}$. Теорема доказана.

В шифраторах обычно используются операции и преобразования над векторным пространством $V_m(p)$, полем Галуа $GF(p^m)$ и кольцом вычетов Z_{p^m}. В качестве следствий теоремы 1 приведем группу $\Pi_{W,W}$ для этих случаев.

Следствие 1. Пусть $H \in \{GF(p^m), V_m(p)\}$. Для любого подпространства $W<H$, $\dim W = t \in \{\overline{1, m-1}\}$, множество $\Pi_{W,W}$ является импримитивной группой из $S(H)$ с системой импримитивности $\{\beta+W | \beta \in H\}$. Группа $\Pi_{W,W}$ подобна группе $S_{p^t} \int S_{p^{m-t}}$.

Следствие 2. Пусть W – идеал кольца вычетов Z_{p^m}, $|W| = p^t$, $t \in \{\overline{1, m-1}\}$. Тогда множество $\Pi_{W,W}$ является импримитивной группой из $S(Z_{p^m})$ с системой импримитивности $\left\{\beta+W \mid \beta \in Z_{p^m}\right\}$. Группа $\Pi_{W,W}$ подобна группе $S_{p^t} \int S_{p^{m-t}}$.

2. Описание множества $\Pi_{\alpha,\gamma}$

Для произвольного модуля H над кольцом вычетов приведем описание множества $\Pi_{\alpha,\alpha}$.

Теорема 2. Пусть α – произвольный ненулевой элемент Z_l-модуля H, $|H| = b$, $l \geq 2$ – произвольное натуральное число, $d = |<\alpha>|$. Тогда множество $\Pi_{\alpha,\alpha}$ является импримитивной группой из $S(H)$ с системой импримитивности $\{\beta+<\alpha> | \beta \in H\}$. Группа $\Pi_{\alpha,\alpha}$ подобна группе $Z_d \int S_{d^{-1}b}$.

Доказательство. Аналогично теореме 1 показывается, что множество $\Pi_{\alpha,\alpha}$ является группой.

Поскольку $(\alpha+\beta)^{\pi} = \alpha+\beta^{\pi}$ для любого элемента $\beta \in H$, то справедливо равенство $(\alpha c+\beta)^{\pi} = \alpha c+\beta^{\pi}$ для любого элемента $\alpha c \in <\alpha>$, $c \in Z_l$, и любого элемента $\beta \in H$. Значит, $(\beta+<\alpha>)^{\pi} = \beta^{\pi}+<\alpha>$ для любого элемента $\beta \in H$. Следовательно, группа $\Pi_{\alpha,\alpha}$ имеет систему импримитивности $\{\beta+<\alpha> | \beta \in H\}$.

Рассмотрим действие группы $\Pi_{\alpha,\alpha}$ на произвольном блоке импримитивности $<\alpha>+\beta$. В этом случае $\beta^{\pi} \in \beta+<\alpha>$. Пусть $\beta^{\pi} = \alpha r+\beta$ для некоторого элемента $\alpha r \in <\alpha>$. Тогда справедливо равенство $(\alpha c+\beta)^{\pi} = \beta+\alpha(c+r)$ для любого элемента $\alpha c \in <\alpha>$. Таким образом, группа $\Pi_{\alpha,\alpha}$ подобна аддитивной группе кольца вычетов Z_d по сложению. Теорема доказана.

В качестве следствий теоремы 2 приведем множество $\Pi_{\alpha,\alpha}$ для $H \in \{GF(p^m), V_m(p), Z_{p^m}\}$.

Следствие 1. Пусть $H \in \{GF(p^m), V_m(p)\}$. Для любого ненулевого $\alpha \in H$ множество $\Pi_{\alpha,\alpha}$ является импримитивной группой из $S(H)$ с системой импримитивности $\{\beta+<\alpha> | \beta \in H\}$. Группа $\Pi_{\alpha,\alpha}$ подобна группе $Z_p \int S_{p^{m-1}}$.

Следствие 2. Для любого ненулевого $\alpha \in Z_{p^m}$, $|<\alpha>| = p^t$, $t \in \{\overline{1, m-1}\}$, множество $\Pi_{\alpha,\alpha}$ является импримитивной группой из $S(Z_{p^m})$ с системой импримитивности $\left\{\beta+<\alpha> \mid \beta \in Z_{p^m}\right\}$. Группа $\Pi_{\alpha,\alpha}$ подобна группе $Z_{p^t} \int S_{p^{m-t}}$.

Для произвольного модуля H над кольцом вычетов покажем, что множество $\Pi_{\alpha,\gamma}$ является правым смежным классом группы $S(H)$ по подгруппе $\Pi_{\alpha,\alpha}$.

Утверждение 1. Пусть α, γ – произвольные ненулевые элементы Z_l-модуля H, $\alpha \neq \gamma$, $d = |<\alpha>|$, $|H| = b$, $l \geq 2$ – произвольное натуральное число. Тогда

$$\Pi_{\alpha,\gamma} = \begin{cases} \left(\Pi_{\alpha,\alpha}\right)h = \left(Z_d \int S_{d^{-1}b}\right)h, \text{ если } |<\alpha>| = |<\gamma>|, \\ \varnothing, \text{ если } |<\alpha>| \neq |<\gamma>|, \end{cases}$$

где h – произвольное линейное отображение из $\Pi_{\alpha,\gamma}$, $\alpha^h = \gamma$.

Доказательство. Для всех $\beta \in H$, $r \in Z_l$ справедливо равенство $(\alpha r + \beta)^\pi = \beta^\pi + \gamma r$. Если $\Pi_{\alpha,\gamma} \neq \varnothing$ при $|<\alpha>| \neq |<\gamma>|$, то $\beta^\pi = \beta^\pi + \gamma d$ и $\gamma d \neq 0$ при $r = d$, что невозможно. Значит, $\Pi_{\alpha,\gamma} = \varnothing$ при $|<\alpha>| \neq |<\gamma>|$.

Если $|<\alpha>| = |<\gamma>|$, то всегда существует линейное отображение $h \in H$, такое, что $\alpha^h = \gamma$, и $h \in \Pi_{\alpha,\gamma}$. Следовательно, $\Pi_{\alpha,\gamma} \neq \varnothing$.

Рассмотрим теперь случай, когда $|<\alpha>| = |<\gamma>|$, т.е. $\Pi_{\alpha,\gamma} \neq \varnothing$. Очевидно, что $\Pi_{\alpha,\gamma} \cap \Pi_{\alpha,\alpha} = \varnothing$ при $\alpha \neq \gamma$. Если $s \in \Pi_{\alpha,\gamma}$ и $g \in (\Pi_{\alpha,\alpha})s$, то выполняется равенство $(\beta + \alpha)^g = (\beta + \alpha)^{\pi s} = (\beta^\pi + \alpha)^s = \beta^{\pi s} + \gamma$, где $g = \pi s$ для некоторой подстановки $\pi \in \Pi_{\alpha,\alpha}$.

Кроме того, если $s \in \Pi_{\alpha,\gamma}$, то $s^{-1} \in \Pi_{\gamma,\alpha}$, так как равенство $(\beta + \alpha)^s = \beta^s + \gamma$ влечет $(\theta + \gamma)^{s^{-1}} = \theta^{s^{-1}} + \alpha$, где $\theta = \beta^s$.

Для любых подстановок $s_1, s_2 \in \Pi_{\alpha,\gamma}$ из справедливости равенства $(\beta + \alpha)^{s_1 s_2^{-1}} = (\beta^{s_1} + \gamma)^{s_2^{-1}} = \beta^{s_1 s_2^{-1}} + \alpha$ следует, что $s_1 s_2^{-1} \in \Pi_{\alpha,\alpha}$. Таким образом, $\Pi_{\alpha,\gamma} = \Pi_{\alpha,\alpha} h$.

Из теоремы 2 следует, что $\Pi_{\alpha,\alpha} = Z_d \int S_{d^{-1}b}$. Значит, $\Pi_{\alpha,\gamma} = \left(Z_d \int S_{d^{-1}b}\right)h$. Утверждение доказано.

Следствие 1. Пусть α, γ – произвольные ненулевые элементы Z_l-модуля H, $\alpha \neq \gamma$, $|<\gamma>| = d$, $|H| = b$, $l \geq 2$ – произвольное натуральное число, $d = |<\alpha>|$. Тогда

$$\Pi_{\alpha,\gamma} = \begin{cases} h\left(\Pi_{\gamma,\gamma}\right) = h\left(Z_d \int S_{d^{-1}b}\right), \text{ если } |<\alpha>| = |<\gamma>|, \\ \varnothing, \text{ если } |<\alpha>| \neq |<\gamma>|, \end{cases}$$

где h – произвольное линейное отображение из $\Pi_{\alpha,\gamma}$, $\alpha^h = \gamma$.

Доказательство аналогично доказательству утверждения 1.

Следствие 2. Пусть выполнены условия утверждения 1. Тогда $\left|\Pi_{\alpha,\gamma}\right| = (b/d)! \cdot d^{b/d}$, если $|<\alpha>| = |<\gamma>|$, и $|\Pi_{\alpha,\gamma}| = 0$, если $|<\alpha>| \neq |<\gamma>|$.

Следствие 3. Пусть $H \in \{GF(p^m), V_m(p)\}$. Для любых ненулевых элементов $\alpha, \gamma \in H$ множество $\Pi_{\alpha,\gamma} = \Pi_{\alpha,\alpha} h = \left(Z_p \int S_{p^{m-1}}\right)h$, где h – произвольное линейное отображение из $\Pi_{\alpha,\gamma}$. Число подстановок из $S(H)$, обладающих линейным транслятором α, равно $(p^m - 1) \cdot p^{p^{m-1}} \cdot (p^{m-1}!)$.

Доказательство следует из утверждения 1, следствия 2 из него и того, что $|<\alpha>| = |<\gamma>| = p$ для любых ненулевых элементов $\alpha, \gamma \in H \setminus 0$.

Следствие 4. Пусть α, γ – произвольные ненулевые элементы кольца Z_{p^m}, $\alpha \neq \gamma$, $|<\alpha>| = p^t$, $t \in \{\overline{1, m-1}\}$. Тогда

$$\Pi_{\alpha,\gamma} = \begin{cases} \left(\Pi_{\alpha,\alpha}\right)h = \left(Z_{p^t} \int S_{p^{m-1}}\right)h, \text{ если } |<\alpha>| = |<\gamma>|, \\ \varnothing, \text{ если } |<\alpha>| \neq |<\gamma>|, \end{cases}$$

где h – произвольное линейное отображение из $\Pi_{\alpha,\gamma}$, $\alpha^h = \gamma$.

Число подстановок из $S(Z_{p^m})$, обладающих линейным транслятором α, равно

$$(p^{m-t} - p^{m-t-1}) \cdot p^{t \cdot p^{m-t}} \cdot (p^{m-t}!).$$

Доказательство следует из утверждения 1, следствия 2 из него и того, что число элементов $\gamma \in Z_{p^m}$, удовлетворяющих равенству $|<\gamma>| = p^t$ в Z_{p^m}, равно $\varphi(p^{m-t}) = p^{m-t} - p^{m-t-1}$.

Для $H \in \{GF(p^m), V_m(p)\}$ покажем, что множества $\bigcap_{i=1}^{k} \Pi_{<\alpha_i>,<\alpha_i>}$, где $\{\alpha_1, \ldots, \alpha_k\} \subseteq H \setminus 0$, $k \geq 1$, также являются сплетениями групп подстановок. Кроме того, подобные группы возникают при описании силовских p-подгрупп группы $S(H)$.

Теорема 3. Пусть $H \in \{GF(p^m),\ V_m(p)\}$. Для любого подмножества $\{\alpha_1, \ldots, \alpha_k\} \subseteq H$, $k \in \overline{\{1, p^m\}}$, $\dim<\alpha_1, \ldots, \alpha_k> = t \geq 1$, имеет место равенство $\bigcap_{i=1}^{k} \Pi_{<\alpha_i>,<\alpha_i>} = \underbrace{S_p \int \ldots \int S_p}_{t} \int S_{p^{m-t}}$.

Доказательство. Обозначим $G^{(1)} = \bigcap_{i=1}^{k} \Pi_{<\alpha_i>,<\alpha_i>}$. Аналогично доказательству теоремы 1 показывается, что $G^{(1)}$ является группой.

Предположим, что элементы $\alpha_1, \ldots, \alpha_t \in H$ линейно независимы. Пусть $\gamma \in H \backslash 0$. Для произвольного множества $X = \{x_1, \ldots, x_r\}$, $r \geq 1$, обозначим $X+\gamma = \{x_1+\gamma, \ldots, x_r+\gamma\}$.

Положим $X_\gamma^{(0)} = \gamma$, $X_\gamma^{(1)} = \gamma + <\alpha_1, \ldots, \alpha_t>$ для $\gamma \in H$, $[X^{(1)}] = \{X_\gamma^{(1)} \mid \gamma \in H\}$, $X_\gamma^{(2)} = \gamma + <\alpha_2, \ldots, \alpha_t>$ для $\gamma \in X_0^{(1)}$, $[X^{(2)}] = \{X_\gamma^{(2)} \mid \gamma \in X_0^{(1)}\}$, $\ldots$, $X_\gamma^{(t-1)} = \gamma + <\alpha_t>$ для $\gamma \in X_0^{(t-2)}$, $[X^{(t-1)}] = \{X_\gamma^{(t-1)} \mid \gamma \in X_0^{(t-2)}\}$, $X_\gamma^{(t)} = \gamma$ для $\gamma \in X_0^{(t-1)}$.

Непересекающиеся множества вида $X_\gamma^{(1)}$, $\gamma \in H \backslash 0$, образуют максимальную систему импримитивности группы $G^{(1)}$, а $X_\gamma^{(j)}$, $\gamma \in X_0^{(j-1)}$ – максимальную систему импримитивности группы $G^{(1)}$ при ограничении ее действия на множество $X_0^{(j-1)}$, $j = \overline{2, t-1}$.

Для $j = \overline{1, t-1}$ рассмотрим естественный гомоморфизм $\varphi_j : G^{(j)} \to \overline{G}^{(j)}$ импримитивной группы $G^{(j)}$ с блоками импримитивности $X_\gamma^{(j)}$ с $\operatorname{Ker}\varphi_j = G^{(j+1)}$. Нетрудно убедиться, что $\operatorname{Im}\varphi_j = S([X^{(j)}])$ и $G^{(j)} = G^{(j+1)} \int \overline{G}^{(j)}$. Кроме того, $S([X^{(j)}]) = S_{p^{m-t}}$ при $j = 1$ и $S([X^{(j)}]) = S_p$ при $j \geq 2$, $G^{(t)} = S_p$. Значит, $G^{(1)} = G^{(2)} \int S_{p^{m-t}} = G^{(3)} \int S_p \int S_{p^{m-t}} = \ldots = S_p \int \ldots \int S_p \int S_{p^{m-t}}$. Теорема доказана.

Теорема 4. Пусть $H \in \{GF(p^m),\ V_m(p)\}$. Для любого подмножества $\{\alpha_1, \ldots, \alpha_k\} \subseteq H$, $k \in \overline{\{1, p^m\}}$, $\dim <\alpha_1, \ldots, \alpha_k> = t \geq 1$, имеет место равенство $\bigcap_{i=1}^{k} \Pi_{\alpha_i,\alpha_i} = \underbrace{Z_p \int \ldots \int Z_p}_{t} \int S_{p^{m-t}}$.

Доказательство аналогично доказательству теоремы 3.

Утверждение 2. Пусть W_1 – произвольный нетривиальный подмодуль R-модуля H, $W_1 = <\alpha_1, \ldots, \alpha_t>$, $t \geq 1$, $I = W_1 + \alpha$, $\alpha \in H \backslash W_1$, $|H| = b$, $|W_1| = d_1$. Пусть также $W_2 = <\alpha_1, \ldots, \alpha_t, \alpha>$, $|W_2| = d_2$. Тогда множество $\Pi_{I,I}$ является импримитивной группой из $S(H)$ с системой импримитивности $\{\beta + W_2 \mid \beta \in H\}$. Группа $\Pi_{I,I}$ подобна группе $S_{d_1} \int S_{d_1^{-1}d_2} \int S_{d_2^{-1}b}$.

Доказательство следует из теоремы 1.

ЛИТЕРАТУРА

1. *Evertse J.H.* Linear structures in block ciphers // EUROCRYPT '87. Springer Verlag,1987.
2. *Chaum D.*, *Evertse J.H.* Cryptanalysis of DES with a reduced number of rounds sequences of linear factors in block ciphers // Crypto '85. Springer Verlag, 1985.
3. *Meier W.*, *Staffelbach O.* Nonlinearity criteria for cryptographic functions // EUROCRYPT'89. Springer Verlag, 1989.
4. *Погорелов Б.А.*, *Пудовкина М.А.* Линейные структуры групп подстановок векторных пространств // Труды 3-й Междунар. конф. «Проблемы безопасности и противодействия терроризму, 2007». М.: МЦНМО, 2008.
5. *Логачев О.А.*, *Сальников А.А.*, *Ященко В.В.* Булевы функции в теории кодирования и криптологии. М.: МЦНМО, 2004.

2008 Теоретические основы прикладной дискретной математики № 1(1)

DOI 10.17223/20710410/1/6

УДК 519.728

ЭНТРОПИЯ НЕДООПРЕДЕЛЕННЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ПРИ ОГРАНИЧЕНИЯХ НА ДООПРЕДЕЛЕНИЯ

Л.А. Шоломов

Институт системного анализа РАН, г. Москва

E-mail: sholomov@isa.ru

Рассматриваются последовательности недоопределенных символов, каждому из которых соответствует некоторое множество полностью определенных символов, одним из которых он может быть замещен (доопределен). При заданных ограничениях на вид доопределний получены оценки минимальной мощности доопределяющего множества для класса последовательностей с заданными кратностями появления символов.

Ключевые слова: *недоопределенный символ*, *доопределение*, *комбинаторная энтропия*, *ограниченная комбинаторная энтропия*, *W-энтропия.*

Задано множество $M = \{0, 1, \ldots, m-1\}$ и каждому непустому $T \subseteq M$ сопоставлен символ a_T. Алфавит всех символов a_T обозначим через A, а его подалфавит $\{a_0, a_1, \ldots, a_{m-1}\}$, символы которого соответствуют элементам множества M, – через A_0. Символы из A_0 будем называть *основными*, из A – *недоопределенными*. *Доопределением* символа a_T назовем всякий основной символ a_i, $i \in T$, а доопределением последовательности в алфавите A – любую последовательность в алфавите A_0, полученную из исходной заменой всех символов некоторыми доопределениями. Будем рассматривать последовательности длины n в алфавите A. Для набора натуральных чисел $\boldsymbol{n} = (n_T, T \subseteq M)$, $\sum_T n_T = n$, обозначим через $K_n(\boldsymbol{n})$ множество всех последовательностей длины n, в которых символы a_T, $T \subseteq M$, встречаются n_T раз. Скажем, что некоторое множество последовательностей в алфавите A_0 *доопределяет* класс $K_n(\boldsymbol{n})$, если в нем найдется доопределение каждой последовательности из этого класса. Пусть $N_n(\boldsymbol{n})$ – минимальная мощность множества, доопределяющего $K_n(\boldsymbol{n})$. Величину $\log N_n(\boldsymbol{n})$ (здесь и дальше логарифмы двоичные) будем называть *комбинаторной энтропией* класса $K_n(\boldsymbol{n})$.

Введем обобщение энтропийной функции Шеннона на случай недоопределенных данных (подробнее см. в [1]), положив для всякого набора $P = (p_T, T \subseteq M)$, $p_T \geq 0$, $\sum_T p_T = 1$:

$$H(P) = \min_Q \left\{ -\sum_{T \subseteq M} p_T \log \sum_{i \in T} q_i \right\},$$

где минимум берется по наборам $Q = (q_i, i \in M)$, $q_i \geq 0$, $\sum_i q_i = 1$. Для полностью определенных данных, когда множествам T мощности $|T| \geq 2$ соответствуют $p_T = 0$, функция $H(P)$ совпадает с обычной энтропийной функцией $H(p_0, \ldots, p_{m-1}) = -\sum_{0 \leq i \leq m-1} p_i \log p_i$. Введем функционал $h_n(\boldsymbol{n}) = n\, H(\boldsymbol{n}/n)$, где $\boldsymbol{n}/n = (n_T/n, T \subseteq M)$. В [1] получены следующие оценки комбинаторной энтропии.

Утверждение 1. Существует константа $c = c(m)$, такая, что комбинаторная энтропия класса $K_n(\boldsymbol{n})$ заключена в пределах

$$h_n(\boldsymbol{n}) - c \log n \leq \log N_n(\boldsymbol{n}) \leq h_n(\boldsymbol{n}) + c \log n.$$

В ряде случаев возникают ограничения на вид используемых доопределений (см., например, [2]). Для набора натуральных чисел $\boldsymbol{s} = (s_0, \ldots, s_{m-1})$, $s_0 + \ldots + s_{m-1} = n$, обозначим через $K_n(\boldsymbol{s})$ класс всех последовательностей из $(A_0)^n$, в которых символы a_i, $i \in M$, встречаются s_i раз. Считаем, что набор $\boldsymbol{s}$ *совместим* с $\boldsymbol{n}$, если в $K_n(\boldsymbol{s})$ имеются доопределения всех (эквивалентно – хотя бы одной) последовательностей из $K_n(\boldsymbol{n})$. Будем говорить, что *множество* $\boldsymbol{S} \subseteq (A_0)^n$ *совместимо с* $\boldsymbol{n}$, если любой набор из $\boldsymbol{S}$ совместим с $\boldsymbol{n}$. Пусть $K_n(\boldsymbol{S}) = \bigcup_{\boldsymbol{s} \in \boldsymbol{S}} K_n(\boldsymbol{s})$. Для $\boldsymbol{S}$, совместимого с $\boldsymbol{n}$, обозначим через $N_n(\boldsymbol{n})_{\boldsymbol{S}}$ минимальное число последовательностей из $K_n(\boldsymbol{S})$, доопределяющих класс $K_n(\boldsymbol{n})$, и величину $\log N_n(\boldsymbol{n})_{\boldsymbol{S}}$ назовем $\boldsymbol{S}$*-ограниченной комбинаторной энтропией* класса $K_n(\boldsymbol{n})$.

В работе [2] приведены оценки $\boldsymbol{S}$-ограниченной энтропии для возникших там случаев – когда множество $\boldsymbol{S}$ одноэлементно (теорема 6) и еще в одной ситуации (теорема 7 без доказательства). В данной работе рас-

сматривается произвольное множество $\boldsymbol{S}$, совместимое с $\boldsymbol{n}$. Чтобы сформулировать основной результат, введем ряд понятий.

Пусть $R = \|r_{Ti}\|$ – матрица, столбцы которой соответствуют множествам $T \subseteq M$, строки – элементам $i \in M$. Элементы матрицы удовлетворяют условиям $r_{Ti} \geq 0$, $r_{Ti} = 0$ для $i \notin T$, $\sum_{T,i} r_{Ti} = 1$. С матрицей R свяжем функцию

$$I(R) = \sum_{T,i} r_{Ti} \log \frac{r_{Ti}}{\sum_T r_{Ti} \sum_i r_{Ti}}.$$

Если рассматривать R как матрицу совместных вероятностей, то $I(R)$ – величина взаимной информации [3]. Пусть заданы наборы $P = (p_T, T \subseteq M)$, $p_T \geq 0$, $\sum_T p_T = 1$, и некоторое множество $\boldsymbol{Q}$ наборов $Q = (q_i, i \subseteq M)$, $q_i \geq 0$, $\sum_i q_i = 1$. Будем говорить, что *матрица R согласована с P при ограничениях $\boldsymbol{Q}$*, если $\sum_i r_{Ti} = p_T$ и набор $Q = (q_0, \ldots, q_{m-1})$, $q_i = \sum_T r_{Ti}$, содержится в $\boldsymbol{Q}$. Величину $\inf_R I(R)$ по всем матрицам R, согласованным с P при ограничениях $\boldsymbol{Q}$, обозначим через $H(P)_{\boldsymbol{Q}}$. В частности, если множество $\boldsymbol{Q}$ состоит из одного набора Q, будем использовать обозначение $H(P)_Q$. Значение $H(P)_Q$ достижимо, и вместо $\inf_R$ в этом случае можно использовать $\min_R$. Отметим, что функции $H(P)_W$ подобного типа для различного рода ограничений W на вид совместных распределений возникают в задачах кодирования с заданным критерием верности и носят название W-энтропии [4].

Для множества $\boldsymbol{S}$, согласованного с $\boldsymbol{n}$, введем функционал $h_n(\boldsymbol{n})_{\boldsymbol{S}} = n\, H(\boldsymbol{n}/n)_{\boldsymbol{S}/n}$, где $\boldsymbol{S}/n$ означает множество наборов $\boldsymbol{s}/n$, $\boldsymbol{s} \in \boldsymbol{S}$. Если $\boldsymbol{S}$ состоит из одного набора $\boldsymbol{s}$, будем использовать обозначение $h_n(\boldsymbol{n})_{\boldsymbol{s}}$. Так как число наборов длины n конечно и величина $h_n(\boldsymbol{n})_{\boldsymbol{s}}$ достижима, то $h_n(\boldsymbol{n})_{\boldsymbol{S}} = \min\limits_{\boldsymbol{s} \in \boldsymbol{S}} h(\boldsymbol{n})_{\boldsymbol{s}}$. Сформулируем основной результат работы.

Теорема. Существует константа $c = c(m)$, такая, что для любых $\boldsymbol{n}$ и $\boldsymbol{S}$, где $\boldsymbol{S}$ согласовано с $\boldsymbol{n}$, $\boldsymbol{S}$-ограниченная комбинаторная энтропия класса $K_n(\boldsymbol{n})$ заключена в пределах

$$h_n(\boldsymbol{n})_{\boldsymbol{S}} - c \log n \leq \log N_n(\boldsymbol{n})_{\boldsymbol{S}} \leq h_n(\boldsymbol{n})_{\boldsymbol{S}} + c \log n.$$

Для доказательства понадобится следующий факт.

Лемма. Если для целочисленных наборов $\boldsymbol{n} = (n_T, T \subseteq M)$ и $\boldsymbol{s} = (s_i, i \in M)$, таких, что $\sum_T n_T = \sum_i s_i$, система уравнений относительно переменных w_{Ti}

$$\sum_i w_{Ti} = n_T \ (T \subseteq M), \qquad \sum_T w_{Ti} = s_i \ (i \in M), \qquad w_{Ti} = 0 \ (i \notin T) \tag{*}$$

имеет неотрицательное решение, то для всякого такого решения $\boldsymbol{w} = (w_{Ti}, T \subseteq M, i \in M)$ найдется целочисленное неотрицательное решение $\boldsymbol{w}^0 = (w^0_{Ti}, T \subseteq M, i \in M)$, для которого $\left|w^0_{Ti} - w_{Ti}\right| \leq 1$ при всех T и i.

Доказательство. Рассмотрим решение $\boldsymbol{w}$ и положим $u_{Ti} = \lfloor w_{Ti} \rfloor$, $w'_{Ti} = w_{Ti} - u_{Ti}$, $n'_T = \sum_i w'_{Ti}$, $s'_i = \sum_T w'_{Ti}$. Очевидно, что числа n'_T и s'_i будут целыми и что выполнены уравнения, полученные из (*) заменой всех величин штрихованными, а также условие $\sum_T n'_T = \sum_i s'_i$. Построим ориентированную потоковую сеть [5] с полюсами s (источник) и t (сток), внутренними вершинами α_T, $T \subseteq M$, и β_i, $i \in M$, дугами (s, α_T) с пропускными способностями n'_T, дугами (α_T, β_i), $i \notin T$, с пропускными способностями 1 и (β_i, t) с пропускными способностями s'_i. Поток в этой сети не превосходит $\sum_T n'_T = \sum_i s'_i$, и эта величина достигается при назначении потоков в дугах (α_T, β_i) равными w'_{Ti}. По теореме Форда – Фалкерсона в этой сети существует максимальный целочисленный поток. Обозначив через w''_{Ti} $(w''_{Ti} \in \{0,1\})$ соответствующие потоки в дугах (α_T, β_i) и положив $w^0_{Ti} = u_{Ti} + w''_{Ti}$, получим величины w^0_{Ti}, которые, как нетрудно видеть, удовлетворяют условиям леммы.

Доказательство теоремы. Н и ж н я я о ц е н к а. Обозначим через $t_{\boldsymbol{S}}(\boldsymbol{n})$ максимальное число последовательностей из $K_n(\boldsymbol{n})$, доопределимых одной последовательностью из $K_n(\boldsymbol{S})$. Рассмотрим последовательности $\boldsymbol{x} \in K_n(\boldsymbol{n})$ и $\boldsymbol{y} \in K_n(\boldsymbol{S})$, такие, что $\boldsymbol{x}$ доопределяет $\boldsymbol{y}$. Пусть $\boldsymbol{y} \in K_n(\boldsymbol{S})$, $\boldsymbol{s} = (s_i, i \in M)$. Обозначим через w_{Ti} число символов a_T в $\boldsymbol{x}$, доопределенных в $\boldsymbol{y}$ символом a_i. Числа w_{Ti} удовлетворяют условиям (*). При заданных w_{Ti} последовательность $\boldsymbol{y}$ доопределяет

$$\frac{s_0!}{\prod\limits_T w_{T0}!} \cdots \frac{s_{m-1}!}{\prod\limits_T w_{T,m-1}!} = \frac{\prod\limits_i s_i!}{\prod\limits_{T,i} w_{Ti}!}$$

последовательностей из $K_n(\boldsymbol{n})$, а всего она доопределяет

$$t_{\boldsymbol{s}}(\boldsymbol{n}) = \sum_{\{w_{Ti}\},(*)} \frac{\prod_i s_i!}{\prod_{T,i} w_{Ti}!}$$

последовательностей из этого класса, где сумма берется по всем наборам неотрицательных чисел w_{Ti}, удовлетворяющих условиям (*). В силу $0 \le w_{Ti} \le n$ и того, что количества индексов i и множеств T ограничены константами (зависящими от m), имеем

$$t_{\boldsymbol{s}}(\boldsymbol{n}) \le n^{c_1} \max_{\{w_{Ti}\},(*)} \frac{\prod_i s_i!}{\prod_{T,i} w_{Ti}!},$$

где $c_1 = c_1(m)$ – константа. Отсюда

$$t_{\boldsymbol{S}}(\boldsymbol{n}) = \max_{\boldsymbol{s}\in\boldsymbol{S}} t_{\boldsymbol{s}}(\boldsymbol{n}) \le n^{c_1} \max_{\boldsymbol{s}\in\boldsymbol{S}} \max_{\{w_{Ti}\},(*)} \frac{\prod_i s_i!}{\prod_{T,i} w_{Ti}!}.$$

Класс $K_n(\boldsymbol{n})$ содержит $n!\Big/\prod_T n_T!$ последовательностей. Отсюда и из предшествующего неравенства заключаем, что

$$N_n(\boldsymbol{n})_{\boldsymbol{S}} \ge \frac{|K_n(\boldsymbol{n})|}{t_{\boldsymbol{S}}(\boldsymbol{n})} \ge n^{-c_1} \min_{\boldsymbol{s}\in\boldsymbol{S}} \min_{\{w_{Ti}\},(*)} \frac{n!\prod_{T,i} w_{Ti}!}{\prod_T n_T! \prod_i s_i!}.$$

Из формулы Стирлинга следует, что для любых целых $z, z_1, \ldots, z_k$, $z \ge 2$, $z_1 + \ldots + z_k = z$, выполнено $\log\left(z!\Big/\prod_j z_j!\right) = z\log z - \sum_j z_j \log z_j + \theta \log z$, $-c_2 \le \theta \le c_2$, $c_2 = c_2(k)$. С учетом этого получаем

$$\log N_n(\boldsymbol{n})_{\boldsymbol{S}} \ge \min_{\boldsymbol{s}\in\boldsymbol{S}} \min_{\{w_{Ti}\},(*)} \left(n\log n - \sum_T n_T \log n_T - \sum_i s_i \log s_i + \sum_{T,i} w_{Ti} \log w_{Ti} \right) - c\log n.$$

Используя равенства $\sum_T n_T = \sum_i s_i = \sum_{T,i} w_{Ti} = n$ и (*), преобразуем минимизируемое выражение к виду

$$n\left(-\sum_{T,i} \frac{w_{Ti}}{n} \log \frac{w_{Ti}}{n} + \sum_T \frac{n_T}{n} \log \frac{n_T}{n} + \sum_i \frac{s_i}{n} \log \frac{s_i}{n} \right) = n\,\mathrm{I}\left(\left\| \frac{w_{Ti}}{n} \right\| \right),$$

откуда

$$\log N_n(\boldsymbol{n})_{\boldsymbol{S}} \ge \min_{\boldsymbol{s}\in\boldsymbol{S}} \min_{\{w_{Ti}\},(*)} \left\{ n\,\mathrm{I}\left(\left\| \frac{w_{Ti}}{n} \right\| \right) \right\} - c\log n \ge$$

$$\ge \min_{\boldsymbol{s}\in\boldsymbol{S}} nH(\boldsymbol{n}/n)_{\boldsymbol{s}/n} - c\log n = \min_{\boldsymbol{s}\in\boldsymbol{S}} h_n(\boldsymbol{n})_{\boldsymbol{s}} - c\log n = h_n(\boldsymbol{n})_{\boldsymbol{S}} - c\log n.$$

В е р х н я я о ц е н к а. Рассмотрим значение $\boldsymbol{s}$, при котором $h_n(\boldsymbol{n})_{\boldsymbol{s}} = h_n(\boldsymbol{n})_{\boldsymbol{S}}$. Занумеруем последовательности $\boldsymbol{x} \in K_n(\boldsymbol{n})$ и $\boldsymbol{y} \in K_n(\boldsymbol{s})$ индексами $r = 1, 2, \ldots, n!\Big/\prod_T n_T!$ и $q = 1, 2, \ldots, n!\Big/\prod_i s_i!$ соответственно. Образуем таблицу $\|d_{qr}\|$, строки которой соответствуют последовательностям $\boldsymbol{y}_q$, столбцы – последовательностям $\boldsymbol{x}_r$, положив $d_{qr} = 1$ в случае, когда $\boldsymbol{y}_q$ доопределяет $\boldsymbol{x}_r$, и $d_{qr} = 0$, когда нет. Все столбцы таблицы содержат одинаковое число единиц; обозначим его $b_{\boldsymbol{s}}(\boldsymbol{n})$. Оценим $b_{\boldsymbol{s}}(\boldsymbol{n})$ снизу.

Пусть $R = \|r_{Ti}\|$ – матрица, на которой достигается значение $H(\boldsymbol{n}/n)_{\boldsymbol{s}/n}$. Числа $w_{Ti} = nr_{Ti}$ удовлетворяют уравнениям (*). По лемме найдется целочисленное решение $\left\{w^0_{Ti}\right\}$ уравнений (*), компоненты которого отличаются от w_{Ti} не более чем на 1. Для произвольного $\boldsymbol{x}_r = K_n(\boldsymbol{n})$ число доопределений $\boldsymbol{y}_q \in K_n(\boldsymbol{s})$, в которых число символов a_T, доопределяемых символом a_i, равно w^0_{Ti}, составляет $\prod_T n_T!\Big/\prod_{T,i} w^0_{Ti}!$. Поэтому $b_{\boldsymbol{s}}(\boldsymbol{n}) \ge \prod_T n_T!\Big/\prod_{T,i} w^0_{Ti}!$.

В матрице из 0 и 1 с u строками и v столбцами, содержащей не менее s единиц в каждом столбце, можно выделить множество из не более $\frac{u}{s}(\ln\frac{vs}{u}+1)+1$ строк, содержащих 1 в каждом столбце [6]. Из этого результата при

$$u = \frac{n!}{\prod_i s_i!}, \qquad v = \frac{n!}{\prod_T n_T!}, \qquad s = \frac{\prod_T n_T!}{\prod_{T,i} w_{Ti}^0!}$$

получаем оценку

$$N_n(\boldsymbol{n})_{\boldsymbol{S}} \le N_n(\boldsymbol{n})_{\boldsymbol{s}} \le c_1 n \frac{n! \prod_{T,i} w_{Ti}^0!}{\prod_T n_T! \prod_i s_i!},$$

где $c_1 = c_1(m)$ – некоторая константа. Применение формулы Стирлинга дает

$$\log N_n(\boldsymbol{n})_{\boldsymbol{S}} \le n \log n - \sum_T n_T \log n_T - \sum_i s_i \log s_i + \sum_{T,i} w_{Ti}^0 \log w_{Ti}^0 + c_2 \log n.$$

Нетрудно убедиться непосредственно, что при изменении числа x не более чем на 1 величина $x \log x$ изменяется не более чем на $O(\log x)$. С учетом этого и того, что количества индексов T и i ограничены константами (зависящими от m), перейдем от величин w_{Ti}^0 к $w_{Ti} = nr_{Ti}$:

$$\log N_n(\boldsymbol{n})_{\boldsymbol{S}} \le n \log n - \sum_T n_T \log n_T - \sum_i s_i \log s_i + \sum_{T,i} w_{Ti} \log w_{Ti} + c_3 \log n,$$

изменив константу перед $\log n$. Подобно случаю нижней оценки это неравенство может быть преобразовано к виду

$$\log N_n(\boldsymbol{n})_{\boldsymbol{S}} \le n \mathrm{I}\left(\left\| \frac{w_{Ti}}{n} \right\| \right) + c_3 \log n.$$

Принимая во внимание, что

$$n \mathrm{I}\left(\left\| \frac{w_{Ti}}{n} \right\| \right) = n \mathrm{I}(\|r_{Ti}\|) = nH(\boldsymbol{n}/n)_{\boldsymbol{s}/n} = h_{\boldsymbol{s}}(\boldsymbol{n}) = h_{\boldsymbol{S}}(\boldsymbol{n}),$$

приходим к нужной верхней оценке.

Теорема доказана.

Рассмотрим функцию

$$H(P,Q) = -\sum_T p_T \log \sum_i q_i,$$

участвующую в определении функции $H(P)$, и функцию $H(P)_Q$, введенную выше. Из [1, 2] следует, что $\min_Q H(P, Q) = \min_Q H(P)_Q$ $(= H(P))$ и, более того, минимумы достигаются в одной и той же точке Q_0. Функция $H(P, Q)$ задана явно, в то время как $H(P)_Q$ определена как минимум функции $I(R)$ по матрицам R, удовлетворяющим определенным условиям. В связи с этим возникает вопрос о взаимоотношении функций $H(P, Q)$ и $H(P)_Q$ и возможности замены в теореме функционала $h_n(\boldsymbol{n})_{\boldsymbol{S}}$ вычислительно более простым функционалом $h_n(\boldsymbol{n}, \boldsymbol{S})$, где $h_n(\boldsymbol{n}, \boldsymbol{S}) = \min_{\boldsymbol{s} \in \boldsymbol{S}} h_n(\boldsymbol{n}, \boldsymbol{s})$, $h_n(\boldsymbol{n}, \boldsymbol{s}) = n\, H(\boldsymbol{n}/n, \boldsymbol{s}/n)$.

Утверждение 2. Имеет место неравенство $H(P, Q) \le H(P)_Q$.

Доказательство. Пусть значение $H(P)_Q$ достигается на матрице $R = \|r_{Ti}\|$. Ее элементы удовлетворяют соотношениям $\sum_i r_{Ti} = p_T$, $\sum_T r_{Ti} = q_i$. При заданном T, воспользовавшись для выпуклой функции $f(x) = x \log x$ неравенством Иенсена $\sum_i \alpha_i f(x_i) \ge f\left(\sum_i \alpha_i x_i\right)$ при $\alpha_i = q_i \Big/ \sum_{j \in T} q_j$, $x_i = r_{Ti} / p_T q_i$, получаем с учетом $r_{Ti} = 0$ для $i \notin T$

$$\sum_i r_{Ti} \log \frac{r_{Ti}}{p_T q_i} = p_T \left(\sum_{j \in T} q_j \right) \sum_i \frac{q_i}{\sum_{j \in T} q_j} \frac{r_{Ti}}{p_T q_i} \log \frac{r_{Ti}}{p_T q_i} \ge p_T \log \frac{1}{\sum_{j \in T} q_j}.$$

Суммирование этих неравенств по T дает $H(P)_Q \ge H(P, Q)$. Утверждение доказано.

Из него вытекает, что $h_n(\boldsymbol{n}, \boldsymbol{S}) \le h_n(\boldsymbol{n})_{\boldsymbol{S}}$, и в силу теоремы получаем следующий факт.

Следствие. Существует константа $c = c(m)$, такая, что

$$\log N_n(\boldsymbol{n})_{\boldsymbol{S}} \ge h_n(\boldsymbol{n}, \boldsymbol{S}) - c \log n.$$

Таким образом, более просто вычислимый функционал $h_n(\boldsymbol{n}, \boldsymbol{S})$ может быть использован в нижней оценке $\boldsymbol{S}$-ограниченной комбинаторной энтропии. Следующий пример показывает, что он не применим в качестве верхней оценки.

Пример. Рассмотрим некоторый класс $K_n(\boldsymbol{n})$ *частично определенных последовательностей*, т.е. последовательностей в алфавите $A = \{a_0, \ldots, a_{m-1}, \bullet\}$, где $\bullet$ – н*еопределенный символ*, допускающий доопределение любым символом алфавита $A_0 = \{a_0, \ldots, a_{m-1}\}$ (в прежних обозначениях он совпадает с a_M). Пусть

$\boldsymbol{n} = (n_0, \ldots, n_{m-1}, n_\bullet)$ и доопределения берутся из $K_n(\boldsymbol{s})$, $\boldsymbol{s} = (s_0, \ldots, s_{m-1})$, $s_0 \ge n_0, \ldots, s_{m-1} \ge n_{m-1}$. Удобнее иметь дело с частотами, поэтому положим $\boldsymbol{n}/n = P = (p_0, \ldots, p_{m-1}, p_\bullet)$, $P' = (p_0, \ldots, p_{m-1})$, $\boldsymbol{s}/n = Q = (q_0, \ldots, q_{m-1})$.

Поскольку $\log \sum_i q_i = 0$, функция $H(P, Q)$ приобретает вид $H(P,Q) = -\sum_i p_i \log q_i$. Отсюда, учитывая $\sum_i p_i = 1 - p_\bullet$, по свойству энтропийной функции получаем

$$H(P,Q) = -(1-p_\bullet)\sum \frac{p_i}{1-p_\bullet}\log q_i \ge (1-p_\bullet)H\left(\frac{p_0}{1-p_\bullet},\ldots,\frac{p_{m-1}}{1-p_\bullet}\right) = (1-p_\bullet)H\left(\frac{P'}{1-p_\bullet}\right).$$

Равенство здесь достигается лишь при $Q_0 = P'/(1-p_\bullet)$, поэтому $H(P) = H(P, Q_0) = (1-p_\bullet)H(P'/(1-p_\bullet))$.

В данном случае имеется единственная матрица R, согласованная с P при ограничении Q. Ее ненулевыми элементами являются $r_{ii} = p_i$, $r_{\bullet i} = q_i - p_i$, $0 \le i \le m-1$. Следовательно,

$$H(P)_Q = I(R) = \sum_i p_i \log\frac{p_i}{p_i q_i} + \sum_i (q_i - p_i)\log\frac{q_i - p_i}{p_\bullet q_i} =$$

$$= -\sum_i q_i \log q_i + p_\bullet \sum_i \frac{q_i - p_i}{p_\bullet}\log\frac{q_i - p_i}{p_\bullet} = H(Q) - p_\bullet H\left(\frac{Q-P'}{p_\bullet}\right).$$

С учетом $\sum_i (q_i - p_i) = p_\bullet$ и свойств энтропийной функции оценим разность

$$H(P)_Q - H(P,Q) = -p_\bullet \sum_i \frac{q_i - p_i}{p_\bullet}\log q_i - p_\bullet H\left(\frac{Q-P'}{p_\bullet}\right) \ge 0.$$

Равенство достигается лишь при условии $Q = (Q - P')/p_\bullet$, т.е. при $Q = P'/(1-p_\bullet) = Q_0$. Вычислим

$$H(P)_{Q_0} = H(Q_0) - p_\bullet H\left(\frac{Q_0 - P'}{p_\bullet}\right) = H\left(\frac{P'}{1-p_\bullet}\right) - p_\bullet H\left(\frac{P'}{1-p_\bullet}\right) = H(P).$$

Таким образом, функция $H(P, Q)$ всюду меньше функции $H(P)_Q$, исключая точку Q_0, где $\mathrm{H}(P)_{Q_0} = \mathrm{H}(P, Q_0) = \mathrm{H}(P)$. Применительно к функционалам это означает, что $h_n(\boldsymbol{n}, \boldsymbol{s})$ всюду меньше $h_n(\boldsymbol{n})_{\boldsymbol{s}}$, исключая точку минимума, где их значения совпадают и равны $h_n(\boldsymbol{n})$.

ЛИТЕРАТУРА

1. *Шоломов Л.А.* Сжатие частично определенной информации // Нелинейная динамика и управление. Вып. 4. М.: Физматлит, 2004. С. 385 – 399.
2. *Шоломов Л.А.* О сложности последовательной реализации частичных булевых функций схемами // Дискретный анализ и исследование операций. Сер. 1. 2007. Т. 14. № 1. С. 110 – 139.
3. *Галлагер Р.* Теория информации и надежная связь. М.: Сов. радио, 1974.
4. *Вероятность* и математическая статистика: Энциклопедия. М.: БРЭ, 1999.
5. *Форд Л.*, *Фалкерсон Д.* Потоки в сетях. М.: Мир, 1966.
6. *Сапоженко А.А.*, *Асратян А.С.*, *Кузюрин Н.Н.* Обзор некоторых результатов по задачам о покрытии // Методы дискретного анализа в решении комбинаторных задач. Вып. 30. Новосибирск: ИМ СО АН СССР, 1977. С. 46 – 75.

МАТЕМАТИЧЕСКИЕ МЕТОДЫ КРИПТОГРАФИИ

DOI 10.17223/20710410/1/7

УДК 519.7

ЭЛЕМЕНТЫ ТЕОРИИ ДИФФЕРЕНЦИАЛЬНОГО КРИПТОАНАЛИЗА ИТЕРАТИВНЫХ БЛОЧНЫХ ШИФРОВ С АДДИТИВНЫМ РАУНДОВЫМ КЛЮЧОМ

Г.П. Агибалов

Томский государственный университет

E-mail: agibalov@isc.tsu.ru

Метод дифференциального криптоанализа излагается в общем виде применительно к произвольным итеративным блочным шифрам, в которых блок открытого текста преобразуется в блок шифртекста за несколько раундов с использованием на каждом раунде одной и той же операции преобразования, осуществляемой в зависимости от аддитивного раундового ключа. Для этого вводятся необходимые элементы теории функций на конечных абелевых группах, определяется класс рассматриваемых шифров, устанавливаются необходимые вспомогательные предложения и формулируется алгоритм метода, сочетающий применение дифференциальной характеристики и решение системы полиномиальных уравнений над конечным полем. Все построения иллюстрируются на примере DES.

Ключевые слова: *дифференциальный криптоанализ*, *итеративные блочные шифры*, *функции на конечных абелевых группах*, *полиномиальные уравнения над конечным полем.*

Применительно к шифрам, а здесь мы говорим только о них, дифференциальный криптоанализ используется для построения атак с выбором открытого текста, имеющих целью (полное или частичное) раскрытие ключа шифра. Его название происходит от английского difference – разность и связано с тем, что в нем рассматриваются зависимости не между открытыми и шифртекстами, а между разностями пар открытых текстов и разностями пар соответствующих шифртекстов при фиксированном неизвестном ключе. Со времени выхода в свет первых работ по дифференциальному криптоанализу [1 – 3] появились десятки, если не сотни, публикаций, в которых предлагаются конкретные атаки на конкретные шифры (или другие криптосистемы), разработанные на основе этой его идеи. К сожалению, автору не известно ни одной работы, которая содержала бы изложение дифференциального криптоанализа как метода в общем виде, а именно так, как это принято в вычислительной математике – с определением основных его понятий, с формулировкой и доказательством его базовых теорем, с определением классов шифров, к которым он применим, с формулировкой его алгоритма для какого-либо из этих классов или, хотя бы, четких правил (технологии) разработки такого алгоритма. Ничего подобного, к сожалению, на данный момент в криптографии нет. Известные атаки дифференциального криптоанализа на конкретные шифры носят совершенно частный характер и неприменимы к другим шифрам, даже близким по классу.

В данной работе предпринимается попытка хотя бы частично восполнить этот пробел. Здесь мы рассматриваем дифференциальный криптоанализ применительно к произвольным итеративным блочным шифрам, в которых блок открытого текста преобразуется в блок шифртекста за несколько раундов с применением на каждом раунде одной и той же операции преобразования, осуществляемой в зависимости от аддитивного раундового ключа. Изложению метода в общем виде предшествуют введение необходимых элементов теории функций на конечных абелевых группах, определение класса рассматриваемых шифров, их классификация по свойствам раундовой функции, установление необходимых вспомогательных предложений, а также формулировка альтернативного метода для одного частного случая, а именно для шифров с функцией раунда, разделимой по Фейстелю. Этот частный метод основан на теореме об аддитивном раундовом ключе и фактически повторяет классический подход дифференциального криптоанализа к DES с присущими ему недостатками. Общий же метод дифференциального криптоанализа, сформулированный здесь как алгоритм для всех итеративных блочных шифров с аддитивным раундовым ключом, в своей основе сводится к решению системы полиномиальных уравнений над конечным полем для последнего раунда шифра с известными значениями на его выходе, с известной с ненулевой вероятностью разностью значений на его входе, с неизвестными компонентами раундового ключа и с неизвестными значениями на его входе. Для r-раундового шифра разность на входе r-го раунда берется из $(r-1)$-раундовой характеристики этого шифра. Все построения общего характера иллюстрируются на примере DES, взятого без операций начальной и заключи-

тельной перестановок и без расписания раундовых ключей. Решение систем полиномиальных уравнений над конечным полем и построение многораундовых вероятностных характеристик шифров составляют предмет самостоятельных исследований и в работе не рассматриваются.

Справедливости ради следует заметить, что попытки построения теории дифференциального криптоанализа предпринимались и ранее его авторами в [1, 2] и Д.Р. Стинсоном в его замечательной монографии [4], однако эти попытки ограничились рассмотрением только класса DES-подобных шифров и не привели к созданию единого алгоритма их дифференциального криптоанализа.

1. Функции на конечных абелевых группах

Будем рассматривать функции, которые, как и их аргументы, принимают значения в конечных абелевых группах. Некоторые из независимых переменных, от которых могут зависеть рассматриваемые функции, играют особую роль и называются параметрами. Их особая роль заключается в выборе подфункций заданной функции путем фиксации их значений под знаком функции. Для функций шифрования, например, в роли параметров выступают компоненты ключа. Функция, среди аргументов которой имеются или отсутствуют параметры, называется функцией с параметром или без соответственно. Впредь, в отсутствие специальных оговорок, под функцией подразумевается функция без параметров. Условимся также значения переменных функции, не являющихся параметрами, и значения самой функции называть значениями соответственно на ее входе и на ее выходе. Для простоты записи во всех рассматриваемых группах операция сложения в отсутствие дополнительных оговорок обозначается одинаково – символом +.

Пусть далее A, B и I суть конечные абелевы группы, $t = |A|$, $h: A \to B$ есть функция без параметра и $\eta: A \times I \to B$ есть функция с параметром, принимающим значения в I. Будем представлять последнюю системой функций без параметра – ее подфункций $\eta_i: A \to B$, $i \in I$, где $\eta_i(u) = \eta(u, i)$ для всех $u \in A$ и $i \in I$.

Назовем *разностным множеством* функции h любое множество $D_h(a, b) = \{u \in A: h(u) - h(u - a) = b\}$ для $a \in A$ и $b \in B$. Это название обязано тому, что здесь a есть разность $u - u^*$ элементов u и $u^* = u - a$ на входе h, а b – разность $h(u) - h(u^*)$ соответствующих значений на выходе h. По определению, $D_h(a, b)$ есть множество всех тех значений u аргумента функции h, изменение которых на величину a (т.е. с u на $u^* = u - a$) вызывает изменение значения h на величину b (с $h(u)$ на $h(u^*) = h(u) - b$).

Назовем *тестовым множеством* функции $h: A \to B$ любое множество $T_h(v, v^*, b) = \{d - v: d \in D_h(a, b)\}$ для v, v^* в A, b в B и $a = v - v^*$.

Теорема 1. Для любых v, v^*, k в A и $b \in B$ если $b = h(v + k) - h(v^* + k)$, то $k \in T_h(v, v^*, b)$.

Доказательство. Пусть $a = v - v^*$. Тогда $b = h(v + k) - h(v^* + k) = h(v + k) - h(v + k - a)$, $v + k \in D_h(a, b)$ и $k = (v + k) - v \in T_h(v, v^*, b)$. ■

Ввиду $b \neq b' \Rightarrow D_h(a, b) \cap D_h(a, b') = \varnothing$, при любом $a \in A$ все непустые разностные множества $D_h(a, b)$, $b \in B$, образуют разбиение множества A, и функция $p_h(b|a)$ от b, задаваемая равенством $p_h(b|a) = t^{-1}|D_h(a, b)|$, является распределением вероятностей на B. Ее значение для конкретного $b \in B$ есть вероятность того, что для случайного (т.е. выбранного с вероятностью t^{-1}) u в A и для $u^* = u - a$ будет $h(u) - h(u^*) = b$. Будем называть $p_h(b|a)$ *вероятностью разности b на выходе функции h при условии, что разность на ее входе равна a*.

Если ввести $Q(a) = \{(u, u^*) \in A^2: u - u^* = a\}$ и $Q_h(a, b) = \{(u, u^*) \in A^2: u - u^* = a, h(u) - h(u^*) = b\}$, то легко заметить, что $|Q(a)| = t$, $|Q_h(a, b)| = |D_h(a, b)|$ и, следовательно, $p_h(b|a) = |Q_h(a, b)| / |Q(a)|$, т.е. $p_h(b|a)$ – это доля упорядоченных пар наборов на входе h с фиксированной разностью a, соответствующие которым наборы на выходе h имеют разность b.

Для функции с параметром $\eta: A \times I \to B$, для каждого значения ее параметра $i \in I$, имея $a \in A$ и $b \in B$, можно вычислить $D_{\eta_i}(a, b) = \{u \in A: \eta_i(u) - \eta_i(u - a) = b\}$ и $p_{\eta_i}(b \mid a) = t^{-1} | D_{\eta_i}(a, b)|$. Величину $p_{\eta_i}(b \mid a)$ естественно обозначать $p_\eta(b \mid a, i)$ и называть вероятностью разности b на выходе функции η при условии, что разность на ее входе равна a и ее параметр имеет значение i.

Будем называть η *функцией с аддитивным параметром* (в I), если существуют функции $\varphi: A \to I$ и $\psi: A \times I \to B$, такие, что φ – гомоморфизм групп и $\eta(u, i) = \psi(u, \varphi(u) + i)$ для любых $u \in A$ и $i \in I$; в этом случае пишем $\eta = [\psi, \varphi]$.

Функцию $\eta = [\psi, \varphi]$ называем функцией *с ветвлением* или *без ветвления входа*, если в ней $\psi(u, i)$ зависит от аргумента u существенно или фиктивно соответственно; в последнем случае вместо $\psi(u, i)$ пишем $\psi(i)$ и имеем $\eta(u, i) = \psi(\varphi(u) + i)$ для всех $u \in A$ и $i \in I$. Функция $\eta = [\psi, \varphi]$ называется *разделимой*, если $A = A' \times A''$, $B = B' \times B''$ для некоторых абелевых групп A', A'', B', B'', $\varphi(u) = \varepsilon(R)$, $\psi(u, j) = (\lambda(R), \rho(L, j))$ для некоторых функций $\varepsilon: A'' \to I$, $\lambda: A'' \to B'$, $\rho: A' \times I \to B''$ и любых $u = LR \in A' \times A''$ и $j \in I$; в этом случае для всех $u = LR \in A' \times A''$ имеем $\eta(u, i) = [\psi, \varphi](LR, i) = \psi(LR, \varphi(LR) + i) = (\lambda(R), \rho(L, \varepsilon(R) + i))$ и пишем $\eta = [\lambda, \rho, \varepsilon]$. По определению, здесь ε – гомоморфизм групп. Функция $\eta = [\lambda, \rho, \varepsilon]$ называется *разделимой по Фейстелю*, если $A'' = B'$, λ – тождественная функция, $A' = B''$ и $\rho(L, j) = L + \delta(j)$ для некоторой функции $\delta: I \to A'$ и любых $L \in A'$ и $j \in I$; в этом случае имеем $\eta(LR, i) = (R, L + \delta(\varepsilon(R) + i))$ и пишем $\eta = \langle \delta, \varepsilon \rangle$.

Теорема 2. Если $\eta = [\psi, \varphi]$, то $\forall i \in I(p_\eta(b \mid a, i) = p_\psi(b \mid a\varphi(a))$.

Доказательство. Имеем $\eta_i(u) = \eta(u, i) = \psi(u, \varphi(u) + i)$. Пусть $a = u - u^*$. Тогда ввиду гомоморфности φ можно записать $p_{\eta_i}(b \mid a) = p_{\eta_i}(b \mid u - u^*) = p_\psi(b \mid (u, \varphi(u)+i) - (u^*, \varphi(u^*)+i) = p_\psi(b \mid a\varphi(u - u^*)) = p_\psi(b \mid a\varphi(a))$. ■

Таким образом, для функции с аддитивным параметром вероятность разности на ее выходе определяется однозначно разностью на ее входе и не зависит от значения ее параметра.

Следствие 1. Для функции $\eta = [\psi, \varphi]$ без ветвления входа $\forall i \in I(p_\eta(b \mid a, i) = p_\psi(b \mid \varphi(a))$.

Следствие 2. Для функции $\eta = [\lambda, \rho, \varepsilon]$ с тождественной компонентой λ если $a = a'a'' \in A' \times A''$ и $b = b'b'' \in B' \times B''$, то $\forall i \in I(p_\eta(b \mid a, i) = p_\rho(b'' \mid a''\varepsilon(a''))$. В частности, для функции η, разделимой по Фейстелю, т.е. для $\eta = \langle\delta, \varepsilon\rangle$ будет $\forall i \in I(p_\eta(b \mid a, i) = p_\delta(b'' - a' \mid \varepsilon(a''))$.

2. Итеративные блочные шифры

Пусть далее m, n и r суть натуральные числа, обозначающие соответственно длину раундового ключа, длину блока шифра и количество раундов, $K = \{0, 1\}^m$, $X = \{0, 1\}^n$ и $Y = \{0, 1\}^n$ – множества соответственно раундовых ключей, блоков открытых текстов и блоков шифртекстов. Пусть также $U = \{0, 1\}^n$ и g: $U \times K \to U$ – произвольная функция со свойством *обратимости*: для любых u, v в U и k в K если $u \neq v$, то $g(u, k) \neq g(v, k)$. Это свойство позволяет ввести функцию g^{-1}: $U \times K \to U$ как $g^{-1}(u, k) = v$, если $g(v, k) = u$. Определим также функцию G: $X \times K^r \to Y$, положив $G(x, k_1 \ldots k_r) = y$, где y вычисляется итеративно как $y = u_r$, $u_i = g(u_{i-1}, k_i)$, $i = 1, 2, \ldots, r$ и $u_0 = x$. Назовем ее функцией *r-раундового шифрования*. Под ее знаком k_i называется (*раундовым*) *ключом i-го раунда*. Она обратима: $G(x, k_1 \ldots k_r) = y \Rightarrow x = G^{-1}(y, k_1 \ldots k_r)$, где $G^{-1}(y, k_1 \ldots k_r) = u_0$, $u_{i-1} = g^{-1}(u_i, k_i)$, $i = r, r-1, \ldots, 1$ и $u_r = y$. Вместе с множествами K, X, Y и *обращением* G^{-1} она образует *r-раундовый итеративный блочный шифр* $C = (X, Y, K^r, G, G^{-1})$. Функция g называется функцией *раундового шифрования*, или, для краткости, *раундом* этого шифра.

Пример 1. Классическим примером итеративного блочного шифра является DES (без операций начальной и заключительной перестановок и без расписания раундовых ключей). В нем $n = 64$, $m = 48$, $r = 16$, $U = \{0, 1\}^{32} \times \{0, 1\}^{32}$, $g = (g', g'')$, g': $U \times K \to \{0, 1\}^{32}$, $g'(LR, k) = R$, g'': $U \times K \to \{0, 1\}^{32}$, $g''(LR, k) = L \oplus f(R, k)$, f: $\{0, 1\}^{32} \times K \to \{0, 1\}^{32}$, $f(R, k) = P(S(E(R) \oplus k))$, E: $\{0, 1\}^{32} \to \{0, 1\}^{48}$, S: $\{0, 1\}^{48} \to \{0, 1\}^{32}$, P: $\{0, 1\}^{32} \to \{0, 1\}^{32}$, $\oplus$ – покомпонентное сложение булевых векторов по модулю 2 и E, S, P – операции соответственно расширения, замены и перестановки в раунде DES.

3. Шифры с аддитивным раундовым ключом

Обратим внимание на следующее существенное свойство функции раундового шифрования DES: в ее выражение раундовый ключ входит аддитивно с инъективным образом шифруемого блока, вследствие чего ее значение определяется фактически не блоком и раундовым ключом, взятыми отдельно, но их суммой с операцией сложения ($\oplus$), которая на множестве раундовых ключей определяет структуру аддитивной абелевой группы. Это свойство присуще функциям раундового шифрования большинства современных итеративных блочных шифров, с той лишь разницей, что групповая операция сложения на множестве раундовых ключей в них может отличаться от $\oplus$. Например, в российском шифре ГОСТ 28147-89 в качестве таковой выступает сложение раундовых ключей как целых чисел по модулю 2^{32}. Впредь итеративные блочные шифры, в которых функция раундового шифрования обладает указанным свойством, мы называем шифрами с аддитивным раундовым ключом. Ниже следует их строгое определение.

Предполагая множества U и K аддитивными абелевыми группами, назовем функцию g: $U \times K \to U$ и итеративный блочный шифр C с функцией раундового шифрования g соответственно *функцией* и *шифром с аддитивным раундовым ключом*, если g есть функция с аддитивным параметром в K, т.е. если $g = [\psi, \varphi]$ для некоторых φ: $U \to K$ и ψ: $U \times K \to U$ и, следовательно, $g(u, k) = \psi(u, \varphi(u) + k)$.

Пример. Функция g раундового шифрования в DES является функцией с аддитивным раундовым ключом, разделимой по Фейстелю. Действительно, для нее $g(LR, k) = (R, (L + P(S(E(R) + k))))$, поэтому $g = [\psi, \varphi]$, где $\varphi(LR) = E(R)$, $\psi(LR, j) = (R, P(S(j)))$ для всех $LR \in U$ и $j \in K$, и, кроме того, $g = \langle\delta, \varepsilon\rangle$, где $\delta(j) = P(S(j))$ и $\varepsilon(R) = E(R)$ для всех $j \in K$ и $R \in \{0, 1\}^{32}$.

Следующая теорема является одной из основ для дифференциального криптоанализа итеративных блочных шифров с аддитивным раундовым ключом.

Теорема 3 (об аддитивном раундовом ключе). Для функции раундового шифрования $g = [\psi, \varphi]$: $U \times K \to U$ с аддитивным ключом в K, для любых u, u^* в U, $k \in K$ и $b \in U$ если $b = g(u,k) - g(u^*,k)$, то $0k \in T_\psi(u\varphi(u), u^*\varphi(u^*), b)$.

Доказательство. Имеем $b = \psi(u\varphi(u) + 0k) - \psi(u^*\varphi(u^*) + 0k)$, и требуемое утверждение получаем по теореме 1 при $A = U \times K$, $B = U$, $h = \psi$, $v = u\varphi(u)$, $v^* = u^*\varphi(u^*)$. ■

В случае, если функция $[\psi, \varphi]$ не имеет ветвления входа, это утверждение принимает следующий вид.

Следствие 1 (теорема об аддитивном ключе раунда без ветвления входа). Для функции раундового шифрования $g = [\psi, \varphi]$: $U \times K \to U$ с аддитивным ключом в K и без ветвления входа, для любых u, u^* в U, $k \in K$ и $b \in U$ если $b = g(u, k) - g(u^*, k)$, то $k \in T_\psi(\varphi(u), \varphi(u^*), b)$. ■

Следствие 2 (теорема об аддитивном ключе раунда с разделимой функцией). Для разделимой функции раундового шифрования $g = [\lambda, \rho, \varepsilon]$: $U \times K \to U$, где $U = U' \times U''$, ε: $U'' \to K$, λ: $U'' \to U'$, ρ: $U' \times K \to U''$, для любых L, L^* в U', R, R^* в U'', $u = LR$, $u^* = L^*R^*$, $k \in K$, $b' \in U'$, $b'' \in U''$ и $b = b'b''$ если $b = g(u, k) - g(u^*, k)$, то $0k \in T_\rho(L\varepsilon(R), L^*\varepsilon(R^*), b'')$. ■

Назовем раундом Фейстеля раунд с аддитивной функцией шифрования, разделимой по Фейстелю.

Следствие 3 (теорема об аддитивном ключе раунда Фейстеля). Для разделимой по Фейстелю функции раундового шифрования $g = \langle\delta, \varepsilon\rangle$: $U \times K \to U$, где $U = U' \times U'$, ε: $U' \to K$, δ: $K \to U'$, для любых L, L^*, R, R^* в U', $L' = L - L^*$, $u = LR$, $u^* = L^*R^*$, $k \in K$, $b' \in U'$, $b'' \in U'$, $b = b'b''$ и $b_1 \in U'$ если $b = g(u, k) - g(u^*, k)$ и $b_1 = b'' - L'$, то $k \in T_\delta(\varepsilon(R), \varepsilon(R^*), b_1)$. ■

Следствие 4 (теорема об аддитивном ключе раунда DES). Для функции g раундового шифрования DES в обозначениях из следствия 3 если $b = g(u, k) - g(u^*, k)$ и $b_1 = P^{-1}(b'' - L')$, то $k \in T_S(\varepsilon(R), \varepsilon(R^*), b_1)$. ■

Положим $\psi_w(j) = \psi(w, j)$ для всех $w \in U$ и $j \in K$.

Теорема 4. Для функции раундового шифрования $[\psi, \varphi]$: $U \times K \to U$ с аддитивным ключом в K, для любых w, u, u^* в U, k в K и $b \in U$ если $b = \psi_w(\varphi(u) + k) - \psi_w(\varphi(u^*) + k)$, то $k \in T_{\psi_w}(\varphi(u), \varphi(u^*), b)$.

Доказательство. Утверждение следует непосредственно из теоремы 1 при $A = K$, $B = U$, $h = \psi_w$, $v = \varphi(u)$, $v^* = \varphi(u^*)$. ■

Обратим внимание на то, что в теореме 4, в отличие от теоремы 3 и ее следствий, величина b не обязана быть разностью $g(u, k) - g(u^*, k)$ на выходе раунда, что открывает дополнительные возможности для дифференциального криптоанализа. Изучение этих возможностей, однако, выходит за пределы настоящей работы.

4. Дифференциальный криптоанализ одного раунда с аддитивным ключом

Согласно теореме 3, если раундом $g = [\psi, \varphi]$ с аддитивным раундовым ключом k некоторые блоки u, u^* на входе раунда зашифровываются в блоки на его выходе, разность которых равна b, то вектор $0k$ с искомым ключом k принадлежит тестовому множеству $T_\psi(u\varphi(u), u^*\varphi(u^*), b)$ и может быть найден в результате пересечения нескольких таких множеств, построенных для различных пар блоков на входе раунда. В этом, собственно, и состоит дифференциальный криптоанализ любого 1-раундового блочного шифра подобного рода. Более детально алгоритм такого криптоанализа с целью нахождения раундового ключа выглядит следующим образом.

Алгоритм A_1. Последовательно выбираются пары блоков открытого текста и для каждой выбранной пары (u_t, u_t^*), $t = 1, 2, \ldots$, и соответствующей пары $(g(u_t, k), g(u_t^*, k))$ блоков шифртекста на выходе раунда вычисляются образы $\varphi(u_t)$, $\varphi(u_t^*)$, разности $a_t = u_t\varphi(u_t) - u_t^*\varphi(u_t^*)$ и $b_t = g(u_t, k) - g(u_t^*, k)$, разностное множество $D_\psi(a_t, b_t) = \{0v: v \in K, \psi(0v) - \psi(0v - a_t) = b_t\}$ и тестовое множество $T_t = T_\psi(u_t\varphi(u_t), u_t^*\varphi(u_t^*), b_t) = \{d - u_t\varphi(u_t): d \in D_\psi(a_t, b_t)\}$. Строится пересечение Q тестовых множеств T_t для всех рассматриваемых t. С получением одноэлементного Q процесс заканчивается с результатом k, где $\{0k\} = Q$. Процесс может быть остановлен и при несколько более слабом условии, а именно с получением тестовых множеств, в которых некоторый один элемент $0k$ встречается много чаще других. Часть k в нем и выдается за результат криптоанализа – искомый раундовый ключ.

Очевидным образом данный алгоритм переформулируется на любой из частных случаев раунда, представленных следствиями 1 – 4 из теоремы 3.

В реальных шифрах компоненты ψ и φ функции раундового шифрования $g = [\psi, \varphi]$ строятся обычно в виде суперпозиции некоторых более простых функций, допуская, в частности, возможность эффективного вычисления требуемых для алгоритма разностных и тестовых множеств. Ниже эта возможность демонстрируется на примере DES.

5. Пример: дифференциальный криптоанализ 1- и 3-раундовых DES

Пусть далее L_0R_0, $L_0^*R_0^*$ – два блока открытого текста на входе 1-го раунда DES и для i-го раунда DES с $i \geq 1$:

k_i – фиксированный раундовый ключ,

L_iR_i, $L_i^*R_i^*$ – два блока шифртекста на выходе раунда,

$e_i = E(R_{i-1})$, $e_i^* = E(R_{i-1}^*)$, $x_i = e_i \oplus k_i$, $x_i^* = e_i^* \oplus k_i$, $c_i = S(x_i)$, $c_i^* = S(x_i^*)$, $e_{i1}e_{i2}\ldots e_{i8} = e_i$, $e_{i1}^* e_{i2}^* \ldots e_{i8}^* = e_i^*$, $k_{i1}k_{i2}\ldots k_{i8} = k_i$, $(S_1, S_2, \ldots, S_8) = S$, S_j: $\{0, 1\}^6 \to \{0,1\}^4$, $|e_{ij}| = |e_{ij}^*| = |k_{ij}| = 6$, $e_{ij}' = e_{ij} \oplus e_{ij}^*$, $j = 1, 2, \ldots, 8$.

Таким образом, по определению DES, $L_i = R_{i-1}$, $L_i^* = R_{i-1}^*$, $R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$, $R_i^* = L_{i-1}^* \oplus f(R_{i-1}^*, k_i)$, $P(c_i) = f(R_{i-1}, k_i)$ и $P(c_i^*) = f(R_{i-1}^*, k_i)$ для $i = 1, 2, \ldots, 16$.

Пусть также $L_i'R_i' = L_iR_i \oplus L_i^*R_i^*$ для $i = 0, 1, \ldots, 16$. Очевидно, здесь $L_i' = R_{i-1}'$ для $i \geq 1$.

Криптоанализ 1-раундового DES

Имеем: $R_1 = L_0 \oplus f(R_0, k_1)$, $R_1^* = L_0^* \oplus f(R_0^*, k_1)$, $R_1' = R_1 \oplus R_1^* = L_0 \oplus L_0^* \oplus f(R_0, k_1) \oplus f(R_0^*, k_1) = L_0' \oplus f(R_0, k_1) \oplus f(R_0^*, k_1) = L_0' \oplus P(c_1) \oplus P(c_1^*) = L_0' \oplus P(c_1 \oplus c_1^*)$, $L_0' \oplus R_1' = P(c_1 \oplus c_1^*)$. Положим $b = P^{-1}(L_0' \oplus R_1')$. Тогда $b = c_1 \oplus c_1^* = S(x_1) \oplus S(x_1^*)$ и по следствию 4 из теоремы 3 $k_1 \in T_S(e_1, e_1^*, b)$ для $e_1 = E(R_0)$, $e_1^* = E(R_0^*)$.

Отсюда – следующий алгоритм криптоанализа 1-раундового DES. Последовательно выбираются пары блоков открытого текста и для каждой такой пары $(L_0R_0, L_0^*R_0^*)$ и соответствующей пары блоков шифртекста $(L_1R_1, L_1^*R_1^*)$ вычисляются $e_1 = E(R_0)$, $e_1^* = E(R_0^*)$, $L_0' = L_0 \oplus L_0^*$, $R_1' = R_1 \oplus R_1^*$, $b = P^{-1}(L_0' \oplus R_1')$, $b_1b_2\ldots b_8 = b$, где $|b_1| = \ldots = |b_8| = 4$, разностные множества $D_{S_j}(e_{1j}', b_j)$ и тестовые множества $T_j = T_{S_j}(e_{1j}, e_{1j}^*, b_j) = \{d - e_{1j} : d \in D_{S_j}(e_{1j}', b_j)\}$ для $j = 1, 2, \ldots, 8$. Для каждого такого j строится пересечение Q_j тестовых множеств T_j, вычисленных так для всех выбранных пар блоков открытого текста. С получением одноэлементных Q_j для всех j процесс заканчивается с результатом $k_1 = k_{11}k_{12}\ldots k_{18}$, где $\{k_{1j}\} = Q_j$, $j = 1, 2, \ldots, 8$.

Криптоанализ 3-раундового DES [4]

Цель атаки – найти раундовый ключ k_3.

Имеем: $R_3 = L_2 \oplus f(R_2, k_3) = R_1 \oplus f(R_2, k_3) = L_0 \oplus f(R_0, k_1) \oplus f(R_2, k_3)$ и, аналогично, $R_1^* = L_0^* \oplus f(R_0^*, k_1) \oplus f(R_2^*, k_3)$, поэтому $R_3' = L_0' \oplus f(R_0, k_1) \oplus f(R_2, k_3) \oplus f(R_0^*, k_1) \oplus f(R_2^*, k_3)$. Выберем $R_0 = R_0^*$. Тогда $R_3' = L_0' \oplus f(R_2, k_3) \oplus f(R_2^*, k_3)$, $R_3' \oplus L_0' = f(R_2, k_3) \oplus f(R_2^*, k_3) = P(c_3) \oplus P(c_3^*) = P(c_3 \oplus c_3^*)$. Пусть $b = P^{-1}(L_0' \oplus R_3')$. Тогда $b = c_3 \oplus c_3^* = S(x_3) \oplus S(x_3^*)$ и по теореме 1 $k_3 \in T_S(e_3, e_3^*, b)$. Здесь $R_2 = L_3$, $R_2^* = L_3^*$, поэтому $e_3 = E(L_3)$, $e_3^* = E(L_3^*)$.

Отсюда – следующий алгоритм криптоанализа 3-раундового DES. Последовательно выбираются пары блоков открытого текста $(L_0R_0, L_0^*R_0^*)$, где $R_0 = R_0^*$, и для каждой такой пары и соответствующей пары блоков шифртекста $(L_3R_3, L_3^*R_3^*)$ вычисляются $e_3 = E(L_3)$, $e_3^* = E(L_3^*)$, $L_0' = L_0 \oplus L_0^*$, $R_3' = R_3 \oplus R_3^*$, $b = P^{-1}(L_0' \oplus R_3')$, $b_1b_2\ldots b_8 = b$, где $|b_1| = \ldots = |b_8| = 4$, разностные множества $D_{S_j}(e_{3j}', b_j)$ и тестовые множества $T_j = T_{S_j}(e_{3j}, e_{3j}^*, b_j) = \{d - e_{3j} : d \in D_{S_j}(e_{3j}', b_j)\}$ для $j = 1, 2, \ldots, 8$. Для каждого такого j строится пересечение Q_j тестовых множеств T_j, вычисленных так для всех выбранных пар блоков открытого текста. С получением одноэлементных Q_j для всех j процесс заканчивается с результатом $k_3 = k_{31}k_{32}\ldots k_{38}$, где $\{k_{3j}\} = Q_j$, $j = 1, 2, \ldots, 8$.

6. Вероятность разности на выходе раунда с аддитивным ключом

В соответствии с теоремой 2 вероятность $p_g(b \mid a, k)$ разности b на выходе раунда $g = [\psi, \varphi]$ с фиксированными разностью a на его входе и аддитивным раундовым ключом k не зависит от последнего и равна $p_\psi(b \mid a\varphi(a))$. Мы обозначаем ее $p_g(b \mid a)$. Это есть вероятность разности b на выходе раунда g с аддитивным ключом при заданной разности a на его входе и любом фиксированном значении раундового ключа.

Как уже говорилось, в реальных итеративных блочных шифрах функция раундового шифрования g строится обычно в виде суперпозиции некоторых более простых функций, допуская возможность эффективного вычисления вероятности $p_g(b|a)$ для любых заданных a и b. Ниже эта возможность демонстрируется на примере раунда DES.

Пример [4]. В раунде DES операция замены S является набором функций S_i: $\{0, 1\}^6 \to \{0, 1\}^4$, $i = 1,\ldots,8$, для нее $p_S(b|a) = \prod_{i=1}^{8} p_{S_i}(b_i \mid a_i)$ для $a = a_1a_2\ldots a_8$ и $b = b_1b_2\ldots b_8$ и $p_{S_i}(b_i \mid a_i) = |D_{S_i}(a_i, b_i)| / 64$ для $i = 1,\ldots,8$. Функция $S_1(x_1x_2x_3x_4x_5x_6)$ задается табл. 1.

Таблица 1

x_1x_6	$x_2x_3x_4x_5$							
	0000	0001	0010	0011	0100	0101	0110	0111
00	1110	0100	1101	0001	0010	1111	1011	1000
01	0000	1111	0111	0100	1110	0010	1101	0001
10	0100	0001	1110	1000	1101	0110	0010	1011
11	1111	1100	1000	0010	0100	1001	0001	0111

x_1x_6	$x_2x_3x_4x_5$							
	1000	1001	1010	1011	1100	1101	1110	1111
00	0011	1010	0110	1100	0101	1001	0000	0111
01	1010	0110	1100	1011	1001	0101	0011	1000
10	1111	1100	1001	0111	0011	1010	0101	0000
11	0101	1011	0011	1110	1010	0000	0110	1101

Ее разностные множества $D_{S_1}(a,b)$ для $a = 110100$ и всевозможных $b \in \{0, 1\}^4$ перечислены в табл. 2.

Таблица 2

b	$D_{S_1}(110100,b)$
0000	
0001	000011, 001111, 011110, 011111, 101010, 101011, 110111, 111011
0010	000100, 000101, 001110, 010001, 010010, 010100, 011010, 011011, 100000, 100101, 010110, 101110, 101111, 110000, 110001, 111010
0011	000001, 000010, 010101, 100001, 110101, 110110
0100	010011, 100111
0101	
0110	
0111	000000, 001000, 001101, 010111, 011000, 011101, 100011, 101001, 101100, 110100, 111001, 111100
1000	001001, 001100, 011001, 101101, 111000, 111101
1001	
1010	
1011	
1100	
1101	000110, 010000, 010110, 011100, 100010, 100100. 101000, 110010
1110	
1111	000111, 001010, 001011, 110011, 111110, 111111

Таким образом, $p_{S_1}(0000 \mid 110100) = 0$, $p_{S_1}(0001 \mid 110100) = 1/8$, $p_{S_1}(0010 \mid 110100) = 1/4$ и т.д. Аналогично вычисляются все другие вероятности $p_{S_i}(b_i \mid a_i)$ в выражении для $p_S(b \mid a)$. Например, $D_{S_1}(001100, 1110) = \{000011, 000110, 001010, 001111, 010011, 010111, 011011, 011111, 100001, 100110, 101010, 101101, 110111, 111011\}$ и $p_{S_1}(1110 \mid 001100) = 14/64$.

По определению функции f в DES (пример 1), для любых $a = u \oplus u^*$, b в $\{0, 1\}^{32}$ и $k \in \{0, 1\}^{48}$ ввиду теоремы 2 $p_f(b \mid a) = p_f(b \mid a, k) = p_S(P^{-1}(b) \mid E(u) \oplus k \oplus E(u^*) \oplus k) = p_S(P^{-1}(b) \mid E(a))$. По определению функции $g = (g', g'')$ раундового шифрования в DES (пример 1), для любых разностей $L_{i-1}'R_{i-1}'$ и $L_i'R_i'$ соответственно на входе и на выходе i-го раунда DES ввиду следствия 2 из теоремы 2 получаем $p_g(L_i'R_i' \mid L_{i-1}'R_{i-1}') = p_{g''}(R_i' \mid L_{i-1}'R_{i-1}') = p_f(R_i' \oplus L_{i-1}' \mid R_{i-1}') = p_S(P^{-1}(R_i' \oplus L_{i-1}') \mid E(R_{i-1}'))$.

Обратим внимание на неравномерность в DES распределения $p_{S_1}(b \mid 110100)$, обязанную неравномощности множеств $D_{S_1}(110100,b)$. Следствием неравномерности некоторых распределений вероятностей $p_{S_i}(b_i \mid a_i)$ является неравномерность распределений $p_g(b|a)$ для некоторых разностей a на входе раунда DES.

Именно неравномерность распределений вероятностей некоторых разностей на выходе раунда, возможная при некоторых разностях на его входе (вследствие неравномощности соответствующих разностных множеств функции ψ), служит еще одной основой для дифференциального криптоанализа любого итеративного блочного шифра с аддитивным раундовым ключом и парой $[\psi, \varphi]$ в качестве функции раундового шифрования g.

7. Дифференциальная вероятностная характеристика итеративного блочного шифра с аддитивным раундовым ключом

Пусть далее C есть r-раундовый итеративный блочный шифр с функцией раундового шифрования g: $U \times K \to U$ и с аддитивными раундовыми ключами $k_1, \ldots, k_r$ в K, где k_i – ключ i-го раунда, $i = 1, \ldots, r$. Для любого натурального $t \le r$ последовательность $\chi_t = (a_0', a_1', p_1, a_2', p_2, \ldots, a_t', p_t)$, где все $a_0', a_1', \ldots, a_t'$ принадлежат U, называется *t-раундовой дифференциальной вероятностной характеристикой* шифра C, если для любого $i \ge 1$ в ней $p_i = p_g(a_i' | a_{i-1}')$, т.е. если a_{i-1} и a_{i-1}^* выбраны в U так, что $a_{i-1} - a_{i-1}^* = a_{i-1}'$, и вычислены $a_i = g(a_{i-1}, k_i)$ и $a_i^* = g(a_{i-1}^*, k_i)$, то $a_i - a_i^* = a_i'$ с вероятностью p_i. Произведение $p = p_1 \ldots p_t$ называется *вероятностью характеристики* χ_t.

Смысл данного понятия следующий: если зафиксировать раундовые ключи шифра C и на его вход подать поочередно два блока открытого текста, разность которых равна a_0', то на выходе t-го раунда будут получены два блока шифртекста, разность которых совпадет с a_t' с вероятностью не меньше p. Таким образом, имея характеристику χ_t для $t < r$, криптоаналитик может проводить на шифр C атаку с выбором открытого текста (выбирая пары блоков с разностью a_0'), в которой, не имея доступа к выходу промежуточного (t-го)

раунда, он с вероятностью p знает разности (соответствующих) блоков на нем и это знание может использовать для достижения цели криптоанализа.

Имея две характеристики шифра $\chi_t = (a_0', a_1', p_1, a_2', p_2, \ldots, a_t', p_t)$ и $\chi_s^* = (b_0', b_1', q_1, b_2', q_2, \ldots, b_s', q_s)$, где $a_t' = b_0'$, можно построить третью – $\chi_{t+s} = (a_0', a_1', p_1, a_2', p_2, \ldots, a_{t+s}', p_{t+s}) = (a_0', a_1', p_1, a_2', p_2, \ldots, a_t', p_t, b_1', q_1, b_2', q_2, \ldots, b_s', q_s)$, которая называется *конкатенацией* первых двух и обозначается $\chi_t \circ \chi_s^*$. Таким образом из 1-раундовых характеристик строятся многораундовые, в том числе *оптимальные* – с наибольшей вероятностью.

Примеры. Всякая t-раундовая вероятностная характеристика DES имеет вид $(L_0'R_0', L_1'R_1', p_1, L_2'R_2', p_2, \ldots, L_t'R_t', p_t)$. В ней для любого $i \geq 0$: $L_i' \in \{0, 1\}^{32}$, $R_i' \in \{0, 1\}^{32}$ и $L_{i+1}' = R_i'$. Приведем несколько примеров характеристик DES, представляя элементы в $\{0, 1\}^{32}$ целыми числами в 16-ричной системе.

1. Последовательность $\alpha = (L_0'R_0', L_1'R_1', p_1)$, где L_0' – любое, $R_0' = 00000000_{16} = L_1'$, $R_1' = L_0'$, $p_1 = 1$, является 1-раундовой характеристикой DES, ибо если $R_0 \oplus R_0^* = R_0'$, то $R_0 = R_0^*$, $R_1' = R_1 \oplus R_1^* = L_0 \oplus f(R_0, k_1) \oplus L_0^* \oplus f(R_0^*, k_1) = L_0 \oplus L_0^* = L_0'$ и $p_g(L_1'R_1' \mid L_0'R_0') = p_f(R_1' \oplus L_0' \mid R_0') = p_S(P^{-1}(R_1' \oplus L_0') \mid E(R_0')) = p_S(P^{-1}(00000000_{16}) \mid E(00000000_{16})) = p_S(00000000_{16} \mid 000000000000_{16}) = 1 = p_1$.

2. Последовательность $\beta = (L_0'R_0', L_1'R_1', p_1)$, где $L_0' = 00000000_{16}$, $R_0' = 60000000_{16} = L_1'$, $R_1' = 00808200_{16}$, $p_1 = 14/64$, является также 1-раундовой характеристикой DES, ибо $P_g(L_1'R_1' \mid L_0'R_0') = p_S(P^{-1}(R_1' \oplus L_0') \mid E(R_0')) = p_S(P^{-1}(00808200_{16}) \mid E(60000000_{16})) = p_S(\mathrm{E}0000000_{16} \mid 300000000000_{16}) = p_{S_1}(1110 \mid 001100) = 14/64 = p_1$.

3. Конкатенация $\alpha \circ \beta$ предыдущих двух характеристик с $L_0' = 60000000_{16}$ в первой является 2-раундовой характеристикой DES $(L_0'R_0', L_1'R_1', p_1, L_2'R_2', p_2)$, в которой $L_0' = 60000000_{16}$, $R_0' = 00000000_{16} = L_1'$, $R_1' = 60000000_{16}$, $p_1 = 1$, $L_2' = R_1'$, $R_2' = 00808200_{16}$, $p_2 = 14/64$. Ее вероятность есть $p = p_1 p_2 = 14/64$.

8. Шифры над конечным полем

Пусть далее s есть натуральное число, $s \geq 2$ и $F = \{0, 1\}^s$ – конечное поле с числом элементов $q = 2^s$. Для любого натурального t, кратного s, т.е. такого, что $t = sl$ для некоторого целого $l > 1$, элементы множества $\{0, 1\}^t = \{0, 1\}^{sl}$ можно рассматривать как векторы длиной l с компонентами в F, а само это множество как векторное пространство F^l. Таким образом, в случае n и m, кратных s, множества $X = Y = U = \{0, 1\}^n$ и $K = \{0, 1\}^m$ можно представлять как векторные пространства F^ν и F^μ соответственно, где $\nu = n/s$ и $\mu = m/s$, а шифр $C = (X, Y, K^r, G, G^{-1})$ и его функцию раундового шифрования g: $U \times K \to U$ – как соответственно шифр и функцию над полем F, а именно: $C = (F^\nu, F^\nu, (F^\mu)^r, G, G^{-1})$ и g: $F^\nu \times F^\mu \to F^\nu$.

В дальнейшем мы пользуемся этой возможностью без дополнительных оговорок, предполагая, что для шифра с аддитивным раундовым ключом групповая операция сложения векторов в F^ν и F^μ совпадает с их покомпонентным сложением в поле F. Кроме того, мы допускаем возможность задания функции g системой Ω функций g_i: $F^\nu \times F^\mu \to F$, $i = 1, 2, \ldots, \nu$, определяемых для произвольных $u = (u_1 \ldots u_\nu) \in F^\nu$ и $k = (k_1 \ldots k_\mu) \in F^\mu$ как $(g_1(u_1, \ldots, u_\nu, k_1, \ldots, k_\mu) \ldots g_\nu(u_1, \ldots, u_\nu, k_1, \ldots, k_\mu)) = g(u_1, \ldots, u_\nu, k_1, \ldots, k_\mu)$ и задаваемых полиномами над F. Для большинства известных итеративных блочных шифров, в том числе для DES, AES, IDEA, SAFER, LOKI91, Serpent, Redoc и др., реализация этой возможности не представляет сколь-либо заметных трудностей.

9. Уравнения раунда с вероятностной разностью на входе

Рассмотрим ситуацию, когда пары (u, u^*) блоков открытого текста на входе раунда криптоаналитику не доступны ни для выбора, ни для наблюдения, но он знает, что их разность $u - u^*$ с вероятностью $p > 0$ совпадает с блоком u', и может наблюдать на выходе раунда его реакции на них $v = g(u, k)$ и $v^* = g(u^*, k)$. Данная ситуация возникает в дифференциальном криптоанализе многораундового шифра на основе вероятностной характеристики. Для определения ключа раунда в этом случае алгоритм A_1 не годится, так как он предполагает построение тестовых множеств, что невозможно сделать без знания u. Кроме того, останов алгоритма по условию одноэлементности пересечения Q тестовых множеств теперь бессмыслен, потому что если на самом деле $u - u^* \neq u'$, то соответствующее тестовое множество не обязано ни содержать раундовый ключ, ни даже быть непустым.

Вместе с тем описание раунда g системой функций Ω позволяет связать неизвестные компоненты $k_1, \ldots, k_\mu$ его ключа k и неизвестные значения на его входе $u(t) = (u_1(t), \ldots, u_\nu(t))$ и $u^*(t) = (u^*_1(t), \ldots, u^*_\nu(t))$ для любого $t = 1, 2, \ldots, \tau$ с соответствующими известными значениями на его выходе $v(t) = (v_1(t), \ldots, v_\nu(t))$ и $v^*(t) = (v^*_1(t), \ldots, v^*_\nu(t))$ и с известными разностями $u' = (u'_1, \ldots, u'_\nu)$ на его входе следующей системой полиномиальных уравнений над полем F:

$$\begin{gathered} g_i(u_1(t), \ldots, u_\nu(t), k_1, \ldots, k_\mu) = v_i(t), \\ g_i(u^*_1(t), \ldots, u^*_\nu(t), k_1, \ldots, k_\mu) = v^*_i(t), \\ u_i(t) - u^*_i(t) = u'_i,\ i = 1, 2, \ldots, \nu; \\ t = 1, 2, \ldots, \tau. \end{gathered} \tag{1}$$

При $\tau \geq \mu/\nu$ эта система переопределенная: количество уравнений в ней не меньше числа неизвестных. Например, для раунда DES, где $\nu = 32$ и $\mu = 48$, это выполняется уже при $\tau \geq 2$. Условие переопределенности, как известно, является необходимым для единственности решения совместной системы. Данная система, кроме того, вероятностная: уравнения в первых двух строчках выполняются с вероятностью 1, а в третьей – с вероятностью p. Таким образом, p^τ есть вероятность совместности всей системы. Это значит, что только для части $p^\tau \cdot q^\nu$ пар из числа q^ν всех пар $(u(t), u^*(t))$ с заданной разностью $u(t) - u^*(t) = u'$ данная система имеет решение. Для того чтобы это число было не меньше 1, надо иметь $p \geq q^{-\nu/\tau} \geq q^{-\nu^2/\mu}$. Так, в случае DES должно быть $p \geq 2^{-21}$.

10. Дифференциальный криптоанализ многораундового шифра

Постановка задачи. Пусть имеются: r-раундовый шифр C с фиксированными аддитивными раундовыми ключами и его $(r-1)$-раундовая дифференциальная вероятностная характеристика $\chi_{r-1} = (a_0', a_1', p_1, \ldots, a'_{r-1}, p_{r-1})$ с вероятностью $p > 0$. Требуется найти ключ $k = (k_1, \ldots, k_\mu)$ последнего раунда шифра C или его непустую часть.

Рассмотрим два подхода к решению этой задачи: 1 – на основе теоремы об аддитивном раундовом ключе и 2 – на основе решения системы уравнений последнего раунда. Первый применяется в случае, если раундовая функция шифрования в C разделима по Фейстелю, второй – в случае произвольного C.

Алгоритм криптоанализа на основе теоремы об аддитивном раундовом ключе

Итак, пусть функция g раунда в C разделима по Фейстелю, т.е. $g = \langle\delta, \varepsilon\rangle$: $U \times K \to U$, где $U = U' \times U''$, ε: $U' \to K$, δ: $K \to U''$ и $g(LR, k) = R\delta(L + (\varepsilon(R) + k))$ для любых L, R в U' и $k \in K$. В этом случае заданная вероятностная характеристика имеет вид $\chi_{r-1} = (L_0'R_0', L_1'R_1', p_1, \ldots, L'_{r-1}R'_{r-1}, p_{r-1})$.

Алгоритм B_1. Его параметром является натуральное число N.

1. Для каждого $t = 1, 2, \ldots, N$ выполним следующую последовательность действий:

1) выбирается пара блоков $(x(t), x^*(t))$ открытого текста на входе C, таких, что $x(t) - x^*(t) = L_0'R_0'$;

2) на выходе C получается пара соответствующих блоков шифртекста $(y(t), y^*(t)) = (G(x(t), z), (G(x^*(t), z))$, где $z \in K^r$ – неизвестный ключ шифра, и находятся такие $L(t)$, $L^*(t)$, $R(t)$, $R^*(t)$ в U', что $L(t)R(t) = y(t)$ и $L^*(t)R^*(t) = y^*(t)$;

3) вычисляются разности $b(t) = b'(t)b''(t) = y(t) - y(t)^*$ с $b'(t)$, $b''(t)$ в U'' и $b_1(t) = b''(t) - L'_{r-1}$, а также разностное множество $D_\delta(\varepsilon(b'(t)), b_1(t)) = \{j \in K: \delta(j) - \delta(j - \varepsilon(b'(t))) = b_1(t)\}$ и тестовое множество $T_t = T_\delta(\varepsilon(L(t)), \varepsilon(L^*(t)), b_1(t)) = \{j - \varepsilon(L(t)): j \in D_\psi(\varepsilon(b'(t)), b_1(t))\}$.

2. Находится такой элемент $k \in K$, который наиболее часто встречается в тестовых множествах T_t для $t = 1, 2, \ldots, N$. Этот элемент и выдается за результат криптоанализа – искомый раундовый ключ.

Сделаем несколько замечаний к данному алгоритму. В п. 1.3 используется L'_{r-1} – часть разности на входе r-го раунда, известная с вероятностью p, отчего результат криптоанализа будет совпадать с истинным раундовым ключом лишь с некоторой вероятностью, которая тем больше, чем больше вероятность p и число N используемых пар блоков открытого текста. Реализация п. 2 требует применения счетчиков ключей из K в тестовых множествах, что предполагает большие затраты памяти. Алгоритм, подобный данному, невозможен для шифра с произвольной разделимой функцией раунда, с произвольной функцией раунда без ветвления входа и, тем более, с произвольной функцией раунда с аддитивным ключом, так как вычисление тестового множества требует знания блока (или, как в разделимом случае, его части) на входе последнего раунда, который в этих случаях недоступен криптоаналитику.

Алгоритм криптоанализа на основе решения системы уравнений

В отличие от предыдущего, этот алгоритм решает поставленную задачу для любого шифра C с аддитивным раундовым ключом, не требуя больших затрат памяти. В нем предполагается, что заданный шифр C представлен как шифр над полем F и его функция раундового шифрования g: $F^\nu \times F^\mu \to F^\nu$ задана системой функций $\Omega = \{g_i: F^\nu \times F^\mu \to F, i = 1, 2, \ldots, \nu\}$.

Алгоритм B. Пусть $\tau = \lceil \mu/\nu \rceil$.

1. Выбираются пары блоков $(a(1), a^*(1)), (a(2), a^*(2)), \ldots, (a(\tau), a^*(\tau))$ открытого текста на входе C, такие, что $a(t) - a^*(t) = a_0'$ для каждого $t = 1, 2, \ldots, \tau$.

2. На выходе C получаются пары соответствующих блоков шифртекста $(v(1), v^*(1)), (v(2), v^*(2)), \ldots, (v(\tau), v^*(\tau))$.

3. Записывается система уравнений (1) с переменными в F, где $(v_1(t),\ldots,v_\nu(t)) = v(t)$, $(v^*_1(t),\ldots,v^*_\nu(t)) = v^*(t)$, $t = 1, 2, \ldots, \tau$, и $(u'_1, \ldots, u'_\nu) = a'_{r-1}$.

4. Если эта система несовместна, то пп.1 – 3 повторяются с выбором в п. 1 другого набора пар блоков открытого текста; в противном случае

5. Находится решение полученной системы уравнений – возможно, частичное: только для переменных в k. Найденные значения последних и есть результат.

Сделаем несколько замечаний к данному алгоритму. Прежде всего, заметим, что если в последнем пункте алгоритма находится полное решение системы уравнений, то становятся известными пары блоков шифртекста на выходе (r–1)-го раунда, и алгоритм может быть повторен с укороченной характеристикой (a_0', a_1', p_1, …, a'_{r-2}, p_{r-2}) для нахождения ключа предпоследнего раунда. Таким путем можно последовательно найти ключи всех раундов шифра.

Количество циклов из пп. 1 – 3, а следовательно, и количество блоков открытого текста, необходимых алгоритму для достижения результата, зависят от вероятности p используемой характеристики χ_{r-1}: чем она выше, тем меньше это число.

Свойство аддитивности раундового ключа собственно алгоритмом непосредственно не используется, оно нужно только для построения необходимой многораундовой характеристики шифра.

Алгоритм можно применять, очевидно, и тогда, когда системой уравнений (1) описывается не один раунд, но ряд из $l \geq 1$ последних раундов шифра, а вместо χ_{r-1} используется характеристика χ_{r-l} и $\tau = \lceil \mu \cdot l / \nu \rceil$. В этом случае результатом криптоанализа будут ключи этих раундов.

В алгоритме можно использовать одновременно не одну, а несколько дифференциальных характеристик. В этом случае в его п. 3 в качестве (1) берется объединение систем уравнений, записанных для разных характеристик, что повышает результативность алгоритма.

В последнее время методы решения систем уравнений над конечным полем развиваются интенсивно [5]; есть многочисленные примеры их успешного применения в криптоанализе (см., например, [5 – 7]). Какой из методов наиболее эффективен в применении к той или иной конкретной системе уравнений, определяется параметрами последней и соотношениями между ними, такими, как число переменных, уравнений и мономов в системе, ее степень, мощность линеаризационного множества и т.п.

Например, для раунда DES система уравнений (1) с $\tau = 2$ содержит в совокупности 192 уравнения и 176 переменных. Каждое уравнение в ней зависит от 7 переменных, и его степень равна 5. Система состоит из 8 подсистем с 24 уравнениями и 22 переменными в каждой, где 8 уравнений линейные вида $x \oplus y = c$. После исключения переменных по правилу $y = x \oplus c$ в каждой подсистеме остается 16 уравнений и 14 переменных. Множества переменных в различных подсистемах не пересекаются, и решение всей системы сводится к решению каждой ее подсистемы в отдельности, что реально осуществимо даже методом исчерпывающего поиска – перебором всех возможных комбинаций значений переменных в подсистеме.

Выражаю благодарность И.А. Панкратовой, указавшей на некоторые опечатки и ошибки в тексте статьи и исследовавшей алгоритм *B* на примере DES в компьютерном эксперименте [8].

ЛИТЕРАТУРА

1. *Biham E.*, *Shamir A*. Differential cryptanalysis of DES-like cryptosystems / Technical Report. The Weizmann Institute of Science. Department of Applied Mathematics: 1990. 105 p. // J. Cryptology. 1991. V. 4. No. 1. P. 3 – 72.
2. *Biham E.*, *Shamir A*. Differential cryptanalysis of the Data Encryption Standard. Springer Verlag, 1993. 188 p.
3. *Biham E.*, *Shamir A*. Differential cryptanalysis of the full 16-round DES // Lect. Not. Comp. Science. 1993. No. 740. P. 494 – 502.
4. *Stinson D.R.* Cryptography. Theory and Practice. CRC Press, 1995. 434 p.
5. *Агибалов Г.П.* Методы решения систем полиномиальных уравнений над конечным полем // Вестник ТГУ. Приложение. 2006. № 17. С. 4 – 9.
6. *Агибалов Г.П.* Логические уравнения в криптоанализе генераторов ключевого потока // Вестник ТГУ. Приложение. 2003. № 6. С. 31 – 41.
7. *Courtois N.*, *Pieprzyk J.* Cryptanalysis of Block Ciphers with Overdefined Systems of Equations // ASIACRYPT 2002, LNCS 2501. 2002. P. 267 – 287.
8. *Панкратова И.А.* Экспериментальное исследование одного алгоритма дифференциального криптоанализа на примере DES // Прикладная дискретная математика (в печати).

2008 Математические методы криптографии № 1(1)
DOI 10.17223/20710410/1/8
УДК 621.391

КРИПТОСИСТЕМЫ КЛЕТОЧНЫХ АВТОМАТОВ

С.К. Росошек, А.А. Боровков, О.О. Евсютин

Томский государственный университет,
Томский государственный университет систем управления и радиоэлектроники

E-mail: rososhek@list.ru

В работе вводится новый класс симметричных криптосистем, в которых криптографические преобразования реализуются посредством обратимых клеточных автоматов. Приведены два конкретных шифра клеточных автоматов: один с окрестностью Мура, другой – на разбиении.

Ключевые слова: *клеточный автомат*, *окрестность*, *разбиение*, *криптосистема.*

Идея клеточных автоматов была сформулирована независимо Дж. фон Нейманом и К. Цусе в конце 40-х годов.

Клеточные автоматы являются дискретными динамическими системами, поведение которых полностью определяется в терминах локальных зависимостей. Клеточный автомат может мыслиться как стилизованный мир. Пространство представлено равномерной сеткой, каждая ячейка которой, или клетка, содержит несколько битов данных (в простейшем случае может быть один бит); время идет вперед дискретными шагами, а законы мира выражаются единственным набором правил, по которым любая клетка на каждом шаге вычисляет свое новое состояние по состояниям ее близких соседей.

Чтобы синтезировать структуры значительной сложности, необходимо использовать большое количество клеток, а для того, чтобы эти структуры взаимодействовали и существенно эволюционировали, необходимо позволить автомату работать на протяжении большого количества шагов. Изменяя начальные условия, мы будем каждый раз получать совершенно иную картину результата.

В начальном состоянии двумерного клеточного автомата мы имеем двумерную плоскость (будем называть ее полем), разделенную на ячейки (клетки), каждая из которых содержит либо единицу, либо ноль. Задано правило, по которому каждая клетка будет вычислять свое будущее состояние по состояниям соседних клеток. Если предположить, что первоначальное состояние клеточного автомата – это исходный текст, подлежащий процедуре зашифрования, то его конечное состояние будет являться шифртекстом. Правило, которое определяет зависимость будущего состояния каждой клетки от текущего состояния соседних клеток, можно изменять. То есть можно задать несколько правил, которые будут меняться в процессе работы клеточного автомата под управлением ключа шифрования.

В результате можно получить следующие преимущества над классическими шифрами:

- ключевая последовательность не будет участвовать ни в каких преобразованиях с открытым текстом, что сделает невозможным нахождение ключа путем осуществления каких-либо преобразований над открытым текстом;
- каждый бит открытого текста будет зависеть от текущего состояния соседних битов, что сделает невозможным дешифрование части сообщения и значительно усложнит криптоанализ.

1. Шифры двумерных клеточных автоматов с окрестностью Мура

Начальное состояние

Получить начальное состояние клеточного автомата можно следующим образом. Если рассматривать исходный текст как последовательность бит, то заполняем им двумерный массив (лучше квадратный) последовательно способом, описанным ниже.

Задание правил

Так как будущее состояние каждой клетки должно зависеть от текущих состояний ее ближайших соседей, то необходимо определиться, какие именно соседние клетки будут входить в окрестность каждой рассматриваемой клетки. Поскольку в двумерном массиве каждая клетка имеет 8 соседей, за исключением клеток, находящихся на границе массива, то оптимальным выбором будем называть окрестностью клетки восемь ее непосредственных соседей, то есть тех клеток, с которыми она соприкасается в массиве, а не только имеет общую сторону. Такая окрестность в теории клеточных автоматов называется окрестностью Мура.

Используем в качестве ключа случайную последовательность из не менее 16 символов, каждому из которых в кодировке ASCII соответствует 8 бит.

Если перевести все символы ключа в двоичную форму, можно создать следующую зависимость. Номер разряда элемента ключа будет соответствовать определенному соседу каждой клетки, а именно: первый разряд – «северо-западному» (для удобства, при указании сторон клетки будем пользоваться терминами сторон света) соседу, второй – «северному», третий – «северо-восточному», четвертый – «восточному» и т. д (пример изображен на рис. 1). Тогда будет производиться операция «исключающего ИЛИ» (XOR) между значением каждой клетки поля (0 или 1) и значением тех соседних ячеек, у которых соответствующий им бит в элементе (байте) ключа равен единице. То есть, если в первом разряде элемента ключа стоит единица, то операция будет произведена между значением текущей клетки и значением ее «северо-западного» соседа. Если, вдобавок, единица стоит еще и, например, в четвертом разряде элемента ключа, то полученный результат будет участвовать в операции «исключающего ИЛИ» со значением «восточного» соседа клетки. Можно сказать, что элемент (байт) ключа будет указывать каждой клетке, с какими соседями из ее окружения ей следует произвести операцию XOR.

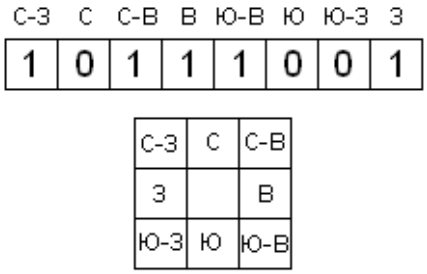


Рис. 1. Принцип соответствия номера бита в байте ключа – пространственному расположению соседей клетки

Таким образом, будет создан клеточный автомат, в котором, как и положено, будущее состояние каждой клетки будет зависеть от текущего состояния своего окружения, и на каждом шаге его работы правило, определяющее эту зависимость, будет меняться в соответствии с введенным ключом. Каждый шаг – новое правило.

Обратимость

Обратимость в теории клеточных автоматов достигается применением специальных обратимых правил.

Обратимости можно добиться, используя плоскость предыдущего состояния, значения клеток в которой учитываются при определении будущего состояния каждой клетки. Но использование этого метода на практике затрудняет необходимость хранения второй плоскости после процедуры зашифрования, так как без нее расшифрование невозможно. Но все же после проведения некоторых исследований стало ясно, что при соблюдении определенных правил работы клеточного автомата можно добиться обратимости и с использованием только одной плоскости.

Было выявлено, что при использовании данного алгоритма обратимость зависит от способа обработки граничных элементов массива, то есть тех элементов, у которых имеется меньше восьми соседей. Можно было просто заполнить их нулями, но в этом случае мы потеряли бы в криптостойкости, так как на обработку граничных элементов некоторые разные элементы ключа влияли бы одинаково. Поэтому нужно добавить недостающих соседей. Было решено заполнить их некоторой константой, генерируемой случайным образом. Чтобы добиться обратимости, эти клетки не должны изменяться на протяжении всего процесса шифрования. Если перед зашифрованием была введена некоторая константа, то перед расшифрованием должна быть введена та же константа, в противном случае расшифрование будет неверным. Независимо от константы ключ должен обеспечивать необходимый уровень защиты, но если у злоумышленника нет той константы, с использованием которой было зашифровано сообщение, это может послужить дополнительной защитой, так как ему придется ее подбирать. Учитывая большой размер блока, эта задача сравнима с перебором всех возможных ключей. Но в случае, если злоумышленник знает константу, которая использовалась при зашифровании сообщения, теоретически это может послужить уязвимостью при проведении криптоанализа. Плюс к этому константу необходимо где-то хранить для дополнительной защиты.

В ходе дальнейшего исследования был найден другой способ, с помощью которого можно добиться обратимости без потери в стойкости и необходимости хранения дополнительных данных. Это замыкание плоскости в торообразную поверхность. В этом случае недостающие соседи граничных клеток берутся с противоположной стороны массива. Но изменение этих «виртуальных» соседей должно осуществляться ди-

намически, то есть как только меняется состояние какой-либо граничной клетки, должно измениться и состояние «виртуального» соседа противоположной клетки на другой стороне массива.

Проблема полного шифрования

В любом блочном шифре существует проблема неполного последнего блока, для решения которой приходится добавлять некоторые лишние символы, чтобы заполнить блок полностью. Как правило, это создает некую уязвимость, которую можно использовать при криптоанализе.

В данном алгоритме нельзя исправить ситуацию таким образом, потому что блок, как правило, имеет очень большой размер, что необходимо для достижения высокой криптостойкости, потому как чем больше размер поля клеточного автомата, тем шире распространяются взаимные зависимости клеток и тем более непредсказуем будет конечный результат. Но благодаря свойствам клеточных автоматов был разработан алгоритм, позволяющий полностью шифровать сообщение без потери в стойкости. Он заключается в том, что сначала определяется такое сочетание размера блока шифрования (размер блока не фиксирован) и их количества, чтобы оставалась незашифрованной часть сообщения, которая может поместиться в блок размером несколько меньше установленного. Эта часть сообщения помещается в дополнительный блок, а именно в центральную часть нового двумерного массива. Перед зашифрованием последнего блока, когда почти все сообщение уже зашифровано, вокруг этой части сообщения, расположенной в центре массива, записываются клетки из последнего шифруемого блока. То есть часть сообщения из последнего шифруемого блока как будто обволакивает ту остаточную часть сообщения, находящуюся в другом блоке. В результате этого граничные клетки дополнительного массива получают соседей из другого блока. Осуществляется один шаг шифрования. Затем, после каждого шага шифрования основного блока дополнительный массив «обволакивается» уже новыми полученными клетками и снова шифруется. Таким образом, на каждом шаге шифрования граничные клетки массива с остаточной частью сообщения будут получать новых соседей из другого блока и эти соседи никак не будут зависеть от значений ячеек данного массива. Благодаря этому, несмотря на меньший размер блока, что удобнее для криптоанализа, злоумышленнику придется подбирать значения соседних клеток для всех граничных клеток массива, да еще и на каждом шаге шифрования, что на практике осуществить невозможно.

Режимы шифрования

После создания первой программной реализации описанного алгоритма стало ясно, что алгоритм работает слишком медленно и не может конкурировать в скорости с лучшими современными криптографическими алгоритмами. Тогда было принято решение реализовать, помимо побитного шифрования, режим побайтного шифрования. В этом случае сам алгоритм не изменится, за исключением того, что каждая клетка двумерного массива (поля клеточного автомата) будет содержать байт, а не бит. Тогда операция XOR производится не с битами, а с байтами.

Описание алгоритма

Генерация ключей

1. Генерируется случайная последовательность символов (алфавит содержит 255 символов), но не менее 16.
2. Каждому символу введенной ключевой последовательности ставится в соответствие уникальное число.
3. Все полученные числа переписываются в двоичном представлении (результат – последовательность нулей и единиц длиной $8N$, где N – количество символов в ключе).

Алгоритм зашифрования

1. Вычисляется сочетание размера и количества блоков, удовлетворяющее следующим условиям:

• Для режима побитного шифрования –

$$(n-2)^2 / 8 \cdot c = f \tag{1}$$

или

$$(n-2)^2 / 8 \cdot c \geq f - \left((n-4)^2 / 8\right), \tag{2}$$

где n – длина блока; c – количество блоков; f – длина сообщения.

Условие (1) выполняется, когда можно подобрать такое сочетание длины и количества блоков, что сообщение можно полностью разбить на такие блоки без остатка. В противном случае должно выполняться условие (2).

• Для режима побайтного шифрования –

$$(n-2)^2 \cdot c = f \tag{3}$$

или

$$(n-2)^2 \cdot c \geq f - (n-4)^2 . \tag{4}$$

Число $(n-2)^2$ в режиме побайтного шифрования (для побитного – $(n-2)^2/8$) равняется количеству символов, помещающихся в один блок. При этом граничные клетки остаются пустыми для того, чтобы записать в них символы константы, или для имитации торообразной поверхности. Числа $(n-2)^2$ и $((n-2)^2/8)$ равняются максимальному количеству остаточных символов, помещающихся в дополнительный блок соответствующего размера. Здесь граничные клетки используются для записи в них значений из последнего основного блока.

2. Каждому символу сообщения ставится в соответствие уникальное число:

• для режима побитного шифрования: число, соответствующее текущему символу, приводится к двоичному виду и записывается в текущие 8 клеток поля, а граничные клетки массива остаются нетронутыми;

• для режима побайтного шифрования: число, соответствующее текущему символу, непосредственно записывается в текущую клетку массива, а граничные клетки массива остаются нетронутыми.

3. Пока блок не будет заполнен полностью, выполняется п. 2.

4. Заполняются граничные клетки массива:

• используется константа:

- режим побитного шифрования: каждому символу сгенерированной константы ставится в соответствие уникальное число; это число приводится к двоичному виду и записывается в 8 граничных клеток;

- режим побайтного шифрования: каждому символу сгенерированной константы ставится в соответствие уникальное число и это число непосредственно записывается в текущую граничную клетку;

• осуществляется замыкание в торообразную поверхность: в каждую граничную клетку записывается значение клетки, расположенной на противоположной стороне двумерного массива (не являющаяся граничной клеткой).

5. Если верно условие (2) или (4) и выполняется зашифрование последнего блока сообщения, то заполняется дополнительный массив для остатка сообщения, который не войдет в основной цикл шифрования.

Выполняются п. 2 – 3 применительно к дополнительному блоку и остатку сообщения.

6. Если справедливо условие (2) или (4) и выполняется зашифрование последнего блока сообщения, то заполняются граничные клетки вокруг заполненных ячеек дополнительного массива, содержащего остаток сообщения, который не вошел в основной цикл шифрования.

Вокруг заполненных ячеек дополнительного массива клетки заполняются значениями соответствующих ячеек из основного блока шифрования.

7. Выполняется преобразование значения текущей клетки (граничные клетки не преобразуются).

Производится операция XOR между значением текущей клетки поля и значениями тех соседних ячеек, у которых соответствующий им бит в элементе ключа равен единице.

Соответствие номера разряда элемента ключа следующее: первый разряд соответствует «северо-западному» соседу, второй – «северному», третий – «северо-восточному», четвертый – «восточному», пятый – «юго-восточному», шестой – «южному», седьмой – «юго-западному», восьмой – «западному».

8. Если шифрование выполняется с замыканием плоскости в торообрзаную поверхность, то после преобразования клетки, имеющей общую сторону с граничной клеткой, полученное значение записывается в граничную клетку на противоположной стороне массива.

9. Пока все клетки массива не прошли преобразование, выполняются п. 7 – 8.

10. Если имеет место условие (2) или (4) и выполняется зашифрование последнего блока сообщения, то снова выполняется п. 6.

11. Если выполняется условие (2) или (4) и производится зашифрование последнего блока сообщения, то выполняется п. 7 применительно к дополнительному массиву, пока все клетки дополнительного массива не пройдут процедуру преобразования.

12. Берется следующий элемент ключевой последовательности.

13. Пока ключевая последовательность не кончится, выполняются п. 7 – 12.

14. Вывод шифртекста:

• для режима побитного шифрования: из массива берутся значения восьми текущих клеток, затем эта последовательность приводится к десятичной форме, а полученное число заменяется символом, который соответствуют этому числу, и записывается в шифртекст, причем значения граничных клеток не берутся;

• для режима побайтного шифрования: из массива берется значение текущей клетки, полученное число заменяется символом, который соответствует этому числу, и записывается в шифртекст, причем значения граничных клеток не берутся.

15. Пока массив не кончится, выполняется п. 14.

16. Если сообщение зашифровано не полностью, осуществляется зашифрование следующего блока сообщения, то есть выполняются п. 2 – 16.

Алгоритм расшифрования

Алгоритм расшифрования полностью идентичен алгоритму зашифрования, за исключением следующих моментов:

- элементы ключа берутся в обратном порядке;
- элементы массива обрабатываются в обратном порядке;
- п. 6 алгоритма зашифрования переносится на место п. 10, то есть первое заполнение граничных клеток дополнительного массива происходит после первого раунда расшифрования основного блока, а не до него.

Очень важным свойством данного криптографического алгоритма является неучастие ключа в преобразованиях сообщения. Обычно ключ участвует в некоторых операциях преобразования с битами шифруемого сообщения, что является потенциальной уязвимостью. На этом основан один из самых успешных методов криптоанализа – линейный криптоанализ. Благодаря популярному в наши дни методу отказа оборудования удается взламывать даже самые стойкие шифры. С его помощью удается выявить следы взаимодействия битов ключа с битами шифруемого сообщения на каком-либо шаге работы алгоритма. Но здесь ключ не участвует в операциях преобразования с битами шифруемого сообщения, он только указывает каждому биту (байту) сообщения, с какими именно соседями производить операцию XOR. Вследствие этого линейный криптоанализ к данному шифру просто неприменим, а метод отказа оборудования, скорее всего, не сможет обнаружить никаких следов взаимодействия битов ключа с битами сообщения, потому как их там просто не будет.

Учитывая все вышесказанное, можно утверждать, что криптосистема, разработанная на базе клеточных автоматов, может получить серьезные преимущества в криптостойкости и, возможно, потребуется разработка новых методов криптоанализа для подобных шифров.

2. Двумерные клеточные автоматы на разбиении

Клеточные автоматы на разбиении характеризуются следующим образом.

1. Решетка клеточного автомата разбита на множество конечных однородных частей – *блоков*.

2. *Правило клеточного автомата* задается таким образом, что оно *рассматривает и изменяет содержимое всего блока*, а не отдельной клетки. Одно и то же правило применяется ко всем блокам. Блоки не зависят друг от друга.

3. *Разбиение меняется на каждом шаге*, чтобы разные блоки могли взаимодействовать между собой [1].

Простейшая схема разбиения, когда блоки имеют размер 2 × 2 клетки, а шаги, в которых блоки, соответствующие четной решетке, чередуются с шагами, использующими нечетную решетку, называется окрестностью Марголуса.

В данной работе рассматривается вариант такой схемы разбиения, когда разбиение ограничивается четной и нечетной решетками, но блоки могут иметь больший размер.

Клеточный автомат на разбиении будет обратимым в том случае, если правило клеточного автомата устанавливает взаимно однозначное соответствие между старым и новым состояниями блока. То есть правило представлено в виде таблицы, состоящей из двух столбцов, в первом из которых записаны все возможные текущие состояния блока, а во втором – соответствующие им новые состояния, при этом второй столбец представляет собой перестановку первого столбца. Тогда для того, чтобы развернуть систему в обратном направлении, необходимо поменять столбцы в таблице местами, учитывая при этом разбиение решетки клеточного автомата.

Таким образом, был получен класс обратимых правил, на основе которых разрабатывается новый шифр.

В ходе работы над созданием шифра были написаны несколько программ, некоторые из них представлены ниже.

В программе «Машина клеточных автоматов» реализованы обратимые клеточные автоматы на разбиении, т.е. тот класс клеточных автоматов, о котором говорилось выше.

Клеточные автоматы, реализованные в программе, имеют следующие характеристики:

- решетка клеточного автомата имеет размер 60 × 60 клеток;
- так же как и в окрестности Марголуса, для разбиения решетки клеточного автомата на блоки используются четная и нечетная решетки;
- блоки имеют размер от 2 × 2 до 5 × 5 клеток.

Содержимое одного блока решетки представляется в виде двоичного числа. Каждая клетка решетки клеточного автомата может принимать два состояния, то есть представляет собой бит данных. Чтобы получить значение содержимого блока, необходимо просмотреть значения всех его клеток слева направо и сверху вниз. Таким образом, определяются все биты искомого двоичного числа от самого старшего до самого младшего.

Правило клеточного автомата задается в виде таблицы, состоящей из двух столбцов. В первом столбце перечислены все возможные состояния блока, числа от 0 до $2^{ab} - 1$ в двоичном представлении (a – высота

блока, b – ширина). Второй столбец является перестановкой первого, причем перестановка эта задается простым циклическим сдвигом первого столбца таблицы правил на определенное число позиций.

Программа «Машина клеточных автоматов» дает возможность пользователю, работающему с ней, проследить развитие конкретного клеточного автомата во времени в течение заданного числа шагов.

В программе «Криптосистема клеточных автоматов на разбиении» реализован уже непосредственно сам шифр на основе клеточных автоматов на разбиении.

Развитие клеточного автомата происходит под управлением следующих параметров:

- высота блоков решетки клеточного автомата (a);
- ширина блоков решетки клеточного автомата (b);
- сдвиг в таблице правил (p);
- число шагов развития клеточного автомата во времени (n).

Сдвиг в таблице правил напрямую зависит от размеров блока. Это число, которое выбирается в интервале $1 \le p \le 2^{a \cdot b} - 1$. Число шагов n не взаимосвязано ни с какими другими параметрами.

Ключ разрабатываемого шифра необходимо выбирать таким образом, чтобы все указанные параметры клеточного автомата входили в его состав.

Конечным результатом работы является блочный симметричный шифр на основе клеточных автоматов на разбиении. Шифрование происходит следующим образом: открытый текст разбивается на блоки одинакового размера, соответствующего размеру решетки клеточного автомата, каждый блок используется в качестве начального состояния решетки клеточного автомата, предварительно он должен быть представлен в двоичном виде, затем происходит развитие клеточного автомата во времени под управлением заданного ключа, шифртекстом будет являться конечное состояние решетки клеточного автомата. Расшифрование происходит аналогичным образом: в качестве начального состояния решетки клеточного автомата используется шифртекст, открытым текстом соответственно будет являться конечное состояние решетки клеточного автомата.

На данный момент были разработаны два шифра на основе клеточных автоматов на разбиении. В качестве примера приведем один из них.

Шифр является блочным симметричным шифром, который построен на основе клеточного автомата на разбиении с размерами блока 3×3 клетки (решетка клеточного автомата имеет размер 60×60 клеток) и имеет следующее описание.

1. *Генерация ключей.*

Ключом является случайно сгенерированная битовая строка размером 256 бит.

2. *Алгоритм зашифрования.*

1) Открытый текст разбивается на отдельные блоки по 450 символов, полученные блоки последовательно зашифровываются с помощью одного и того же ключа.

2) Текущий блок открытого текста представляется в двоичном виде, для этого берется ASCII-код каждого символа блока и переводится из десятичной системы в двоичную.

3) Полученной битовой строкой заполняется решетка клеточного автомата, это будет ее начальным состоянием.

4) Клеточный автомат развивается в течение 256 шагов (число шагов равно длине ключа).

5) На каждом шаге развития клеточного автомата разбиение определяется из ключа: если на данном шаге соответствующий бит ключа равен 0, то разбиение на данном шаге соответствует четной решетке, если 1 – разбиение соответствует нечетной решетке. Сдвиг в таблице правил (параметр клеточного автомата p) в двоичном представлении имеет размер 9 бит и определяется из ключа следующим образом: ключ разбивается на две половины, из 4-х байт слева от условной границы берутся младшие биты, из 4-х справа – старшие биты, таким образом определены 8 бит из 9, а последний 9-й бит определяется с помощью операции XOR между ними.

6) Конечное состояние решетки клеточного автомата является блоком шифртекста, соответствующим текущему блоку открытого текста.

7) Полученный блок шифртекста переводится из двоичного в символьное представление.

8) Шаги 2 – 7 повторяются для всех блоков, на которые разбит открытый текст.

3. *Алгоритм расшифрования.*

1) Шифртекст разбивается на отдельные блоки по 450 символов, полученные блоки последовательно расшифровываются с помощью одного и того же ключа.

2) Текущий блок шифртекста представляется в двоичном виде, для этого берется ASCII-код каждого символа блока текста и переводится из десятичной системы в двоичную.

3) Полученной битовой строкой заполняется решетка клеточного автомата, это будет ее начальным состоянием.

4) Клеточный автомат развивается в течение 256 шагов (число шагов равно длине ключа).

5) Данный этап отличается от соответствующего этапа в алгоритме зашифрования только тем, что ключ необходимо просматривать, начиная с конца, а не с начала.

6) Конечное состояние решетки клеточного автомата является блоком исходного текста, соответствующим текущему блоку шифртекста.

7) Полученный блок исходного текста переводится из двоичного в символьное представление.

8) Шаги 2 – 7 повторяются для всех блоков, на которые разбит шифртекст.

Описанный шифр реализован программно в новой версии компьютерной программы «Криптосистема клеточных автоматов на разбиении».

В заключение заметим, что разработанные нами криптосистемы на базе трехмерных клеточных автоматов не вошли в данную статью из-за ограниченности ее размера.

ЛИТЕРАТУРА

1. *Тоффоли Т.*, *Марголус Н.* Машины клеточных автоматов. М.: Мир, 1991. 280 с.

DOI 10.17223/20710410/1/9
УДК 003.26 519.7

КВАДРАТИЧНЫЕ АППРОКСИМАЦИИ СПЕЦИАЛЬНОГО ВИДА ДЛЯ ЧЕТЫРЕХРАЗРЯДНЫХ ПОДСТАНОВОК В S-БЛОКАХ[1]

Н.Н. Токарева

Институт математики им. С.Л. Соболева СО РАН,
Новосибирский государственный университет

E-mail: tokareva@math.nsc.ru

Рассматриваются квадратичные аппроксимации (булевых функций) специального вида и возможность применения их в нелинейном криптоанализе блочных шифров. Для четырехразрядных подстановок, рекомендованных для использования в S-блоках алгоритмов ГОСТ 28147-89, DES, s^3DES, показано, что почти во всех случаях существуют более вероятные (по сравнению с линейными) квадратичные соотношения специального вида на входные и выходные биты этих подстановок.

Ключевые слова: *квадратичный криптоанализ*, *S-блок*, *k-бент-функция.*

Метод линейного криптоанализа (для блочного шифра FEAL) был предложен М. Мацуи и А. Ямагиши в 1992 г., для блочного шифра DES – М. Мацуи [1] в 1993 г.; в настоящее время этот метод наряду с методом дифференциального криптоанализа [2] считается одним из наиболее эффективных. Большое число работ посвящено различным обобщениям и улучшениям метода линейного криптоанализа.

Общий подход к использованию в линейном криптоанализе нелинейных аппроксимаций предложили в 1996 г. Л. Кнудсен и М. Робшау [3], но он не получил пока должного развития в силу своей общности. В этом направлении можно отметить исследования Т. Шимоямы и Т. Канеко [4], связанные с поиском квадратичных соотношений для конкретных подстановок, использующихся в S-блоках DES; экспериментальные исследования Дж. Накахары и др. [5]; работу Ж. Тапиадора и др. [6] по применению эвристических алгоритмов для поиска хороших нелинейных аппроксимаций (с примерами для S-блоков шифра MARS). Вопросы нелинейных аппроксимаций булевых функций (с использованием их приведенного представления) рассматривались также А.В. Ивановым [7].

Мы рассматриваем [8] квадратичные аппроксимации (булевых функций) специального вида и возможность применения их в нелинейном криптоанализе блочных шифров. В работе [9] (см. также [10]) для каждого целого k, $1 \le k \le m/2$ (m четно), была определена бинарная операция $\langle \cdot,\cdot \rangle_k : Z_2^m \times Z_2^m \to Z_2$ на множестве векторов Z_2^m, которую, исходя из ее свойств, можно считать аналогом скалярного произведения векторов над Z_2. Определение было дано в рамках теоретико-кодового подхода; при этом по существу использовалась классификация Z_4-линейных кодов типа Адамара, полученная Д.С. Кротовым [11, 12]. Аналитически операцию $\langle \cdot,\cdot \rangle_k$ можно задать следующим образом:

$$\langle u,v \rangle_k = \left(\bigoplus_{i=1}^{k} \bigoplus_{j=i}^{k} (u_{2i-1} \oplus u_{2i})(u_{2j-1} \oplus u_{2j})(v_{2i-1} \oplus v_{2i})(v_{2j-1} \oplus v_{2j}) \right) \oplus \langle u,v \rangle ,$$

где $\langle u,v \rangle$ – обычное скалярное произведение векторов $u = (u_1, \ldots, u_m), v = (v_1, \ldots, v_m) \in Z_2^m$ над Z_2. При фиксированном векторе $u \in Z_2^m$ функция $\langle u,v \rangle_k$ от переменных v_1, …, v_m является линейной или квадратичной. На основе операции $\langle \cdot,\cdot \rangle_k$ в работе [9] определяются *k-преобразование Уолша – Адамара* $W_f^{(k)}(u) = \sum_{v \in Z_2^m} (-1)^{\langle u,v \rangle_k \oplus f(v)}$ булевой функции f от m переменных и *k-бент-функция* как функция, для которой $W_f^{(j)}(u) = \pm 2^{m/2}$ для каждого $j = 1, \ldots, k$ и любого $u \in Z_2^m$. Заметим, что 1-бент-функции и бент-функции совпадают, а с ростом параметра k «нелинейные» свойства бент-функции усиливаются (см. подробнее [9] и продолжение исследований k-бент-функций в [13]).

[1] Исследование выполнено при финансовой поддержке интеграционного проекта СО РАН № 35 «Древовидный каталог математических Интернет-ресурсов www.mathtree.ru», Российского фонда фундаментальных исследований (проекты 07-01-00248, 08-01-00671) и Фонда содействия отечественной науке.

Через Δ_m обозначим класс всех функций от $v = (v_1, \ldots, v_m)$ вида $\langle u, \pi(v)\rangle_k$, где π – любая перестановка на m элементах $v_1, \ldots, v_m$, параметры u, k произвольны. В [8] предлагается аппроксимировать неизвестные булевы функции функциями из Δ_m. Класс Δ_m состоит из 2^m (т.е. всех) линейных функций и $\sum_{k=2}^{m/2}\binom{m}{2k}2^{m-k}(2k-1)!!$ (что не превышает числа $2^{m(1+\log m)}$) квадратичных функций от m переменных.

Введем простые обозначения, относящиеся к блочным шифрам: m, m_{key} – четные числа; $P = (p_1, \ldots, p_m) \in Z_2^m$ – открытый текст; $C = (c_1, \ldots, c_m) \in Z_2^m$ – шифртекст; $K = (\kappa_1, \ldots, \kappa_{m_{key}}) \in Z_2^{m_{key}}$ – ключ шифрования; $F(P, K)$ – функция шифрования.

Нами предложены [8] модификации алгоритмов 1 и 2 линейного криптоанализа Мацуи [1]. Основная идея модификаций заключается в использовании (линейных и квадратичных) соотношений вида

$$\langle a, \pi(P)\rangle_i \oplus \langle b, \sigma(C)\rangle_j = \langle d, \tau(K)\rangle_k, \qquad (1)$$

выполняющихся с некоторой вероятностью Pr = (1/2) + ε, где параметр ε, называемый *преобладанием* соотношения, отличен от нуля. Здесь векторы $a, b \in Z_2^m$, $d \in Z_2^{m_{key}}$, перестановки π, σ, τ и целые числа i, j, k выбираются криптоаналитиком с целью максимизировать абсолютное значение преобладания ε (задача такого характера является общей для различных методов статистического криптоанализа). Далее при неизвестном K на основе набранной статистики – пар (P,C), где $C = F(P, K)$, – с учетом параметра ε принимается решение о том, что соотношение $\langle d, \tau(K)\rangle_k = \delta$ (для некоторого δ из Z_2) является верным. С помощью полученного соотношения проводится дальнейший анализ шифра. В [8] приведены формулы для вычисления абсолютных значений преобладаний ε (связанные с вычислением k-коэффициентов Уолша – Адамара) и для расчета надежности алгоритмов. Показано, что использование k-бент-функций в качестве функций шифрования позволяет снижать максимальное абсолютное значение преобладания до его минимального значения, а следовательно, предельно повышать стойкость шифра к рассматриваемым квадратичным аппроксимациям. Приведены свойства аппроксимирующих функций, которые могут быть использованы в квадратичном криптоанализе при согласовании нелинейных раундовых аппроксимаций (см. подробнее [8]).

Известно, что стойкость блочного шифра напрямую зависит от стойкости используемых в нем S-блоков. Целью данной работы является рассмотрение примеров четырехразрядных подстановок для S-блоков алгоритмов ГОСТ 28147-89, DES, s^{3}DES и доказательство того факта, что почти во всех случаях существуют более вероятные (по сравнению с линейными) квадратичные соотношения, аналогичные (1), на входные и выходные биты этих подстановок. Результаты получены с помощью компьютера.

Четырехразрядные подстановки в S-блоках

Пример 1. В книге А.Г. Ростовцева и Е.Б. Маховенко [14] приведена серия экстремальных четырехразрядных подстановок $S^1, \ldots, S^{10}$, рекомендованных для S-блоков стандарта ГОСТ 28147-89. Из каждой подстановки путем умножения ее на аффинные подстановки получается целый класс экстремальных подстановок. Все они были выбраны так, чтобы максимально повысить стойкость шифра к методам линейного и дифференциального криптоанализа.

S^1 = (0,13,11,8,3,6,4,1,15,2,5,14,10,12,9,7)
S^2 = (0,1,9,14,13,11,7,6,15,2,12,5,10,4,3,8)
S^3 = (0,1,11,13,9,14,6,7,12,5,8,3,15,2,4,10)
S^4 = (0,1,2,4,3,5,8,10,7,9,6,13,11,14,12,15)
S^5 = (0,1,11,2,8,6,15,3,14,10,4,9,13,5,7,12)
S^6 = (0,1,11,2,8,3,15,6,14,10,4,9,13,5,7,12)
S^7 = (0,4,11,2,8,6,10,1,14,15,3,9,13,5,7,12)
S^8 = (0,4,11,2,8,3,15,1,14,10,6,9,13,5,7,12)
S^9 = (0,11,15,9,1,5,6,8,3,10,4,12,14,13,7,2)
S^{10}= (0,7,10,14,9,1,13,8,12,2,11,15,3,5,4,6)

Найдем наиболее вероятные квадратичные и линейные зависимости между входными и выходными битами произвольной такой подстановки, используя класс аппроксимирующих функций Δ_4.

Число функций в классе Δ_4 равно 28. Из них 16 – линейные функции, 12 – квадратичные, которые можно перечислить следующим образом:

$$\langle 0101, v_1v_2v_3v_4\rangle_2,\ \langle 0110, v_1v_2v_3v_4\rangle_2,\ \langle 1001, v_1v_2v_3v_4\rangle_2,\ \langle 1010, v_1v_2v_3v_4\rangle_2,$$
$$\langle 0101, v_1v_3v_2v_4\rangle_2,\ \langle 0110, v_1v_3v_2v_4\rangle_2,\ \langle 1001, v_1v_3v_2v_4\rangle_2,\ \langle 1010, v_1v_3v_2v_4\rangle_2,$$
$$\langle 0101, v_1v_4v_2v_3\rangle_2,\ \langle 0110, v_1v_4v_2v_3\rangle_2,\ \langle 1001, v_1v_4v_2v_3\rangle_2,\ \langle 1010, v_1v_4v_2v_3\rangle_2.$$

Двоичному вектору $x = (x_1, x_2, x_3, x_4)$ сопоставим целое число $\tilde{x} = 8x_1 + 4x_2 + 2x_3 + x_4$ от 0 до 15. Рассмотрим соотношения $\langle a, \pi(P)\rangle_i \oplus \langle b, \sigma(C)\rangle_j = 0$, где при $i = 1$ вектору a соответствуют числа от 0 до 15 и тождественная перестановка π (что отвечает всем линейным комбинациям битов P); при $i = 2$ вектору a соответствуют числа 5, 6, 9, 10 и перестановки π = id,(1324),(1342) (что отвечает квадратичным комбинациям битов P). Аналогично при $j = 1$ и 2 выбираем значения для b и σ. При данных условиях функции $\langle a, \pi(\cdot)\rangle_i, \langle b, \sigma(\cdot)\rangle_j$

пробегают все множество функций Δ_4 без повторений. Для произвольной подстановки S рассмотрим таблицу, строки которой занумерованы тройками $(i,\tilde{a},\pi)$, а столбцы – тройками $(j,\tilde{b},\sigma)$, такую, что на пересечении строки и столбца находится преобладание ε соответствующего соотношения, умноженное на 16 (или, другими словами, отклонение числа выполнений этого соотношения от половины). И хотя здесь мы приводим способ построения таблицы для четырехразрядной подстановки, он несложно может быть обобщен на случай произвольной t-разрядной подстановки или преобразования $P \to C$, где P и C имеют разное число битов.

Параметром неквадратичности подстановки S назовем число

$$NQ(S) = \min_{i,j} \min_{\substack{a\neq 0,\ \delta\in Z_2 \\ b\neq 0\ \ \pi,\sigma}} \left| \left\{ P : \langle a, \pi(P)\rangle_i \oplus \langle b, \sigma(C)\rangle_j \neq \delta \right\} \right|.$$

Другими словами, величина $NQ(S)$ равна разности числа 8 и максимальной из абсолютных величин элементов таблицы (кроме элементов первой строки и первого столбца). Соотношение, отвечающее элементу таблицы с абсолютной величиной 8-$NQ(S)$, выполняется с вероятностью $NQ(S)/16$ или $1 - NQ(S)/16$ (т.е. наименее или наиболее вероятно).

Нелинейностью подстановки S называется величина

$$NL(S) = \min_{\substack{a\neq 0,\ \delta\in Z_2 \\ b\neq 0\ \ \pi,\sigma}} \left| \left\{ P : \langle a, \pi(P)\rangle_1 \oplus \langle b, \sigma(C)\rangle_1 \neq \delta \right\} \right|.$$

Величину $NL(S)$ можно получить как разность числа 8 и максимальной из абсолютных величин элементов той части таблицы, которая соответствует только линейным соотношениям входных и выходных битов, т.е. где $i = j = 1$ (кроме нулевых комбинаций). Очевидно, что $NQ(S) \le NL(S)$. Согласно [15], справедливо $NL(S) \le 4$ для любой четырехразрядной подстановки S.

Например, для подстановки S^9 имеем $NL(S^9) = 4$, $NQ(S^9) = 2$ (см. табл. 1). В таблице элементы с абсолютными значениями 4 и 6 выделены полужирным шрифтом и серым цветом соответственно.

Таблица 1

Преобладания для подстановки S^9

ε * 16		j=1 id																j=2 id				j=2 i(1324)				j=2 (1342)			
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	5	6	9	10	5	6	9	10	5	6	9	10
i=1 id	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	-2	2	**4**	0	2	2	-2	-2	**4**	0	2	-2	0	0	**4**	-2	2	0	2	0	**4**	2	0	**4**	0	0
	2	0	-2	0	-2	0	-2	**4**	2	2	0	-2	**4**	2	0	2	0	0	2	2	**-4**	0	0	0	**-4**	**4**	2	-2	0
	3	0	2	2	0	0	-2	-2	0	**4**	-2	2	0	**4**	2	-2	0	0	**-4**	0	0	2	0	**4**	-2	0	2	2	**-4**
	4	0	-2	2	0	0	2	2	**-4**	2	0	**4**	2	-2	0	0	2	**4**	0	2	2	2	-2	0	**4**	0	0	-2	2
	5	0	2	0	-2	0	2	0	-2	0	2	0	-2	**4**	-2	**4**	2	0	2	0	2	**4**	2	0	-2	**4**	0	2	-2
	6	0	0	2	2	**4**	0	2	-2	0	0	-2	-2	0	**4**	2	-2	0	2	0	-2	-2	2	0	0	0	2	0	2
	7	0	0	**4**	0	0	0	0	**4**	2	2	2	-2	-2	-2	2	-2	0	0	2	2	0	-2	0	2	0	-2	0	2
	8	0	0	0	**4**	-2	2	2	2	2	-2	-2	-2	0	0	0	**4**	2	2	-2	-2	-2	**4**	0	2	-2	**4**	0	2
	9	0	0	2	2	2	2	0	0	0	**4**	-2	2	2	-2	**-4**	0	-2	**4**	0	2	0	**4**	0	0	2	0	**4**	2
	10	0	2	**-4**	2	2	0	-2	0	**4**	2	0	2	-2	0	2	0	-2	0	0	2	-2	0	0	2	-2	-2	2	2
	11	0	-2	2	0	2	0	**-4**	2	-2	0	0	2	0	2	2	**4**	-2	-2	-2	2	0	0	0	0	-2	-2	2	-2
	12	0	-2	-2	0	-2	**4**	0	2	0	2	2	0	2	**4**	0	-2	2	2	0	**4**	**4**	2	0	2	2	0	2	0
	13	0	2	0	2	-2	**-4**	2	0	-2	**4**	2	0	0	2	0	2	-2	0	**6**	0	-2	-2	**4**	0	2	0	2	**4**
	14	0	**4**	2	2	-2	2	0	0	-2	-2	0	**4**	0	0	2	-2	2	0	-2	0	0	2	0	2	-2	2	0	0
	15	0	**4**	0	**-4**	2	2	2	2	0	0	0	0	-2	2	-2	2	2	2	0	0	2	-2	**-4**	0	2	-2	**-4**	0
i=2 id	5	0	2	-2	0	2	0	0	-2	2	0	0	-2	0	2	**6**	0	0	0	0	0	0	0	0	0	0	0	0	0
	6	0	0	**4**	0	2	2	2	-2	-2	2	-2	-2	**4**	0	0	0	0	**4**	0	0	2	**4**	0	-2	**4**	2	2	0
	9	0	0	0	**4**	**4**	0	0	0	2	2	-2	2	-2	2	-2	-2	-2	2	0	0	**-4**	2	0	2	-2	0	2	**4**
	10	0	2	-2	0	0	2	-2	0	2	**4**	0	2	2	**-4**	0	2	-2	2	0	**4**	2	2	0	0	2	-2	**4**	0
i=2 (1324)	5	0	0	**-4**	0	0	**4**	0	0	0	**4**	0	0	0	0	**4**	0	0	**4**	0	**4**	2	2	-2	2	2	-2	2	2
	6	0	0	2	-2	-2	2	0	0	0	0	2	-2	**6**	2	0	0	2	0	0	2	**6**	2	2	-2	**4**	2	2	**-4**
	9	0	0	-2	2	0	0	-2	2	**4**	0	2	2	0	**4**	-2	-2	0	-2	0	2	0	0	2	2	-2	0	2	0
	10	0	**4**	0	0	2	-2	-2	-2	**4**	0	0	0	2	-2	2	2	-2	-2	0	0	0	0	2	-2	0	0	2	-2
i=2 (1342)	5	0	-2	0	2	2	**4**	2	0	-2	**4**	-2	0	0	2	0	-2	0	**6**	0	2	0	**4**	-2	2	2	0	2	**4**
	6	0	0	0	0	0	0	0	0	2	-2	2	-2	2	**6**	2	-2	2	-2	0	0	2	0	2	0	0	2	0	-2
	9	0	0	0	0	-2	2	-2	2	2	2	2	2	**4**	0	**-4**	0	0	0	0	**4**	**4**	2	2	0	2	0	**4**	-2
	10	0	2	**4**	2	**4**	-2	0	-2	2	0	-2	0	2	0	-2	0	-2	0	0	-2	-2	2	2	-2	0	2	2	0

Как следует из табл. 1, любые линейные соотношения на входные и выходные биты S^9 выполняются с вероятностью не большей 3/4, тогда как существуют 6 квадратичных соотношений, вероятность которых составляет 7/8. Среди них, например, такие соотношения

$$\langle 1100, c_1c_2c_3c_4\rangle_1 \oplus \langle 0110, p_1p_3p_2p_4\rangle_2 = c_1 \oplus c_2 \oplus p_1p_2 \oplus p_1p_4 \oplus p_2p_3 \oplus p_3p_4 \oplus p_1 \oplus p_4 = 0,$$

$$\langle 1110, c_1c_2c_3c_4\rangle_1 \oplus \langle 0101, p_1p_2p_3p_4\rangle_2 = c_1 \oplus c_2 \oplus c_3 \oplus p_1p_3 \oplus p_1p_4 \oplus p_2p_3 \oplus p_2p_4 \oplus p_1 \oplus p_3 = 0,$$

которые, что интересно, линейны относительно битов шифртекста. Аналогично для S^9 можно выбрать соотношения, линейные относительно битов открытого текста. Этот пример показывает, что использование соотношений вида $\langle a, \pi(P)\rangle_i \oplus \langle b, \sigma(C)\rangle_j = 0$ в составе систем уравнений с неизвестными битами (входными или выходными) может приводить к более вероятным аппроксимациям неизвестных битов, не усложняя при этом решение системы (система по-прежнему может оставаться линейной относительно неизвестных битов).

Все подстановки S^1, …, S^{10} имеют параметр нелинейности *NL*, равный 4, тогда как параметр неквадратичности *NQ* для каждой из них равен 2. В табл. 2 приведены количества наиболее вероятных (квадратичных) соотношений на *P* и *C* для подстановок S^1, …, S^{10} и S^{11}, …, S^{18} (для них также *NL* = 4, см. далее). Кроме того, в табл. 2 указано, сколько среди них соотношений, линейных по битам шифртекста и по битам открытого текста (очевидно, одновременной линейности быть не может, так как все соотношения квадратичные).

Таблица 2

Число квадратичных соотношений на входные и выходные биты подстановок S^1,…,S^{18}, выполняющихся с вероятностью 7/8

S: $NL(S) = 4$	Число квадр. соотн. Pr = 7/8	Линейных по битам C	Линейных по битам P
S^1	4	2	2
S^2	7	2	2
S^3	4	2	2
S^4	5	0	2
S^5	2	1	1
S^6	1	0	1
S^7	4	1	1
S^8	4	1	1
S^9	6	3	1
S^{10}	6	2	2
S^{11}	5	2	1
S^{12}	7	1	4
S^{13}	2	0	0
S^{14}	6	4	0
S^{15}	7	1	4
S^{16}	8	0	0
S^{17}	7	2	4
S^{18}	0	0	0

Пример 2. В книге Б. Шнайера [16] приведены восемь четырехразрядных подстановок, использовавшихся при шифровании методом ГОСТ в приложении для ЦБ РФ, а также в однонаправленной хэш-функции ГОСТ. Все они имеют параметр *NL*, равный 2 (кроме подстановки S^{11}, для которой *NL*=4). Для каждой подстановки имеем *NQ* = 2, и в среднем добавляется 5 – 6 новых наиболее вероятных квадратичных соотношений специального вида на входные и выходные биты каждой подстановки. Данные о подстановке S^{11}, вызывающей особый интерес, приведены в табл. 2.

S^{11} = (4,10,9,2,13,8,0,14,6,11,1,12,7,15,5,3)

Пример 3. Для всех 32 подстановок на 16 элементах, используемых в S-блоках алгоритма DES (см., например, [16]), параметры *NL* и *NQ* совпадают и равны 2. Отметим, что для каждой подстановки добавляется от 0 до 11 (в среднем 4 – 5) новых, наиболее вероятных квадратичных соотношений специального вида на входные и выходные биты.

S^{12} = (8,2,11,13,4,1,14,7,5,15,0,3,10,6,9,12)
S^{13} = (10,5,3,15,12,9,0,6,1,2,8,4,11,14,7,13)
S^{14} = (5,10,12,6,0,15,3,9,8,13,11,1,7,2,14,4)
S^{15} = (3,9,15,0,6,10,5,12,14,2,1,7,13,4,8,11)
S^{16} = (15,0,10,9,3,5,4,14,8,11,1,7,6,12,13,2)
S^{17} = (12,6,3,9,0,5,10,15,2,13,4,14,7,11,1,8)
S^{18} = (13,10,0,7,3,9,14,4,2,15,12,1,5,6,11,8)

Пример 4. Рассмотрим 32 подстановки (см., например, [16]) в S-блоках модифицированного алгоритма s^3DES [17], [18], которые считаются устойчивыми к методам дифференциального и линейного криптоанализа. Среди них только 7 подстановок (это подстановки S^{12}, …, S^{18}) обладают нелинейностью *NL* = 4, для 25 остальных параметр

NL равен 2. Для шести из семи подстановок с нелинейностью $NL = 4$ выполняется $NQ = 2$, и в среднем для каждой такой подстановки имеется около 6 квадратичных соотношений с вероятностью 7/8. И лишь для одной подстановки S^{18} квадратичный криптоанализ не дает новой информации: имеем $NL = NQ = 4$.

Заключение

Мы рассмотрели несколько примеров четырехразрядных подстановок с предельно высокой нелинейностью $NL = 4$. Среди 18-ти таких подстановок лишь для одной нельзя построить более вероятные (по сравнению с линейными) квадратичные соотношения специального вида на входные и выходные биты. Для 13-ти из 17-ти остальных подстановок такие соотношения не только существуют, но среди них есть соотношения линейные относительно битов шифртекста, что по существу может использоваться при решении систем линейных уравнений, возникающих при анализе шифров.

Интересным направлением для дальнейшего исследования (и приложения в криптоанализе) является развитие подхода по использованию аналогов линейных свойств у нелинейных операций $\langle\cdot,\cdot\rangle_k$. Поясним сказанное. Если при фиксированном k всем функциям $\langle u, v\rangle_k \oplus \delta$, где $u \in Z_2^m$, $\delta \in Z_2$, сопоставить их векторы значений длины 2^m, то получится двоичный код A_m^k с параметрами кода Адамара. На этом коде достаточно просто можно определить операцию «сложения» • векторов (отличную от $\oplus$ и зависящую от k), относительно которой код образует абелеву группу, причем эта операция согласована с метрикой Хэмминга $d(\cdot,\cdot)$: для любых $x, y \in A_m^k$ выполняется $d(x, y) = d(x \bullet y^{-1}, 0)$, где $y \bullet y^{-1} = 0$. Это обстоятельство и проявляет «линейные свойства» нелинейного скалярного произведения $\langle\cdot,\cdot\rangle_k$.

ЛИТЕРАТУРА

1. *Matsui M.* Linear cryptanalysis method for DES cipher // Advances in Cryptology – EUROCRYPT'93. Workshop on the theory and application of cryptographic techniques (Lofthus, Norway. May 23–27, 1993). Proc. Berlin: Springer, 1994. P. 386 – 397 (LNCS V. 765).
2. *Biham E.*, *Shamir A.* Differential cryptanalysis of DES-like cryptosystems // J. Cryptology. 1991. V. 4. No. 1. P. 3 – 72.
3. *Knudsen L.R.*, *Robshaw M.J.B.* Non-linear approximation in linear cryptanalysis // Advances in Cryptology – EUROCRYPT'96. Workshop on the theory and application of cryptographic techniques (Saragossa, Spain. May 12–16, 1996). Proc. Springer Verlag, 1996. P. 224 – 236 (LNCS V. 1070).
4. *Shimoyama T.*, *Kaneko T.* Quadratic relation of S-box and its application to the linear attack of full round DES // Advances in Cryptology – CRYPTO'98, 18th Annual International Cryptology Conference. (Santa Barbara, California. USA. August 23 – 27, 1998). Proc. Springer, 1998. P. 200 – 211 (LNCS V. 1462).
5. *Nakahara J.*, *Preneel B.*, *Vandewalle J.* Experimental Non-Linear Cryptanalysis // COSIC Internal Report. Katholieke Universiteit Leuven. 2003. 17 p.
6. *Tapiador J. M. E.*, *Clark J. A.*, *Hernandez-Castro J. C.* Non-linear Cryptanalysis Revisited: Heuristic Search for Approximations to S-Boxes // 11th IMA International Conference (Cirencester, UK. December 18 – 20, 2007). Springer, 2007. P. 99 – 117 (LNCS V. 4887).
7. *Иванов А.В.* Использование приведенного представления булевых функций при построении их нелинейных аппроксимаций // Вестник ТГУ. Приложение. 2007. № 23. С. 31 – 35.
8. *Токарева Н.Н.* О квадратичных аппроксимациях в блочных шифрах // Пробл. передачи информ. 2008. Т. 44. № 3. С. 105 – 127.
9. *Токарева Н.Н.* Бент-функции с более сильными свойствами нелинейности: k-бент-функции // Дискр. анализ и исслед. операций. 2007. Сер. 1. Т. 14. № 4. С. 76 – 102.
10. *Tokareva N.N.* On k-bent functions // Вестник ТГУ. Приложение. 2007. № 23. С. 74 – 76.
11. *Кротов Д.С.* Z_4-линейные совершенные коды // Дискрет. анализ и исслед. операций. Сер. 1. 2000. Т. 7. № 4. С. 78–90.
12. *Krotov D.S.* Z_4-linear Hadamard and extended perfect codes // Proc. of the Int. Workshop on Coding and Cryptography WCC 2001, Jan. 8 – 12, 2001. Paris, France, 2001. P. 329 – 334.
13. *Токарева Н.Н.* Описание k-бент-функций от четырех переменных // Дискр. анализ и исслед. операций. 2008. В печати.
14. *Ростовцев А.Г.*, *Маховенко Е.Б.* Введение в теорию итерированных шифров. СПб.: НПО «Мир и Семья», 2003.
15. *Heys H.M.*, *Tavares S.E.* Substitution-permutation networks resistant to differential and linear cryptanalysis // J. Cryptology. 1996. V. 9. No. 1. P. 1 – 19.
16. *Шнайер Б.* Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. М.: Триумф, 2002.
17. *Kim K.*, *Park S.*, *Lee S.* Reconstruction of s2DES S-Boxes and their Immunity to Differential Cryptanalysis // Korea – Japan Workshop on Information Security and Cryptography. (Seoul, Korea. October 24 – 26, 1993): Proc. P. 282 – 291.
18. *Biham E.*, *Biryukov A.* How to strengthen DES using existing hardware // Advances in Cryptology – ASIACRYPT '94, 4th International Conference on the Theory and Applications of Cryptology. (Wollongong, Australia. November 28 – December 1, 1994): Proc. Springer, 1995. P. 398 – 412 (LNCS V. 917).

2008 Математические методы криптографии № 1(1)
DOI 10.17223/20710410/1/10
УДК 621.394

КОМБИНАТОРНО-ГЕОМЕТРИЧЕСКИЕ ПОДХОДЫ К ПОСТРОЕНИЮ СХЕМ ПРЕДВАРИТЕЛЬНОГО РАСПРЕДЕЛЕНИЯ КЛЮЧЕЙ (ОБЗОР)[1]

А.В. Черемушкин

Институт криптографии, связи и информатики Академии ФСБ России, г. Москва

E-mail: avc238@mail.ru

В обзоре рассматривается одна из схем предварительного распределения ключей, называемая схемой на основе пресечений множеств или схемой на основе шаблонов. Приводятся известные результаты об их свойствах и указываются способы построения таких схем.

Ключевые слова: *блок-схема, схема предварительного распределения ключей, схема пересечений множеств.*

Схемы предварительного распределения ключей на основе пресечений множеств являются очень простыми в использовании, не требующими реализации сложных алгебраических операций. Вместе с тем задача их построения оказывается достаточно трудной комбинаторной задачей. Примечательно, что конструкция таких схем явилась источником для применения целого ряда комбинаторных и геометрических понятий. Ниже будут приведены различные известные подходы к построению таких схем.

1. Основные определения

Под схемой предварительного распределения ключей обычно понимают два алгоритма. Первый определяет значения распределяемых между n участниками некоторых значений, которые будем называть ключевыми материалами, а второй позволяет каждой паре (или группе) участников вычислить значение ключа для организации закрытого сеанса. При этом должно выполняться условие, гарантирующее, что никакая другая группа участников, не включающая первую в качестве подмножества, объединив свои ключевые материалы, не сможет получить никакой информации о ключе.

При $n = 2$ достаточно передать каждому участнику один и тот же общий ключ. Поэтому далее предполагаем, что $n > 2$.

Приведем более точные определения.

Пусть $\boldsymbol{U} = \{1, \dots, n\}$ – множество участников; $\boldsymbol{P}, \boldsymbol{F} \subseteq 2^{\boldsymbol{U}}$, $\boldsymbol{P} \cap \boldsymbol{F} = \emptyset$, K – множество ключей.

Определение. *Схема предварительного распределения ключей* (*key predistribution scheme*) ($\boldsymbol{P}, \boldsymbol{F}$)-KPS – это пара множеств, называемых группами привилегированных – $\boldsymbol{P}$ и непривилегированных участников – $\boldsymbol{F}$, такая, что для любой группы $P \in \boldsymbol{P}$ каждый участник i, входящий в эту группу, может вычислить ключ $k_P \in K$, но никакая группа непривилегированных участников не сможет не только восстановить значение k_P, но не сможет получить никакой информации об этом ключе.

Если $\boldsymbol{P}$ состоит из всех подмножеств множества $\boldsymbol{U}$ мощности g, а $\boldsymbol{F}$ – из всех подмножеств мощности w, то будем использовать обозначение (g, w)-KPS. Это такие схемы предварительного распределения ключей, в которых каждая группа из g участников является привилегированной и может сформировать ключ k_P, но ни одна группа F из w участников, отличная от P, не может получить никакой информации об этом ключе. Понятно, что должно выполняться неравенство $g \geq w$.

Эффективность (g, w)-KPS оценивается величиной, которая называется *информационная скорость* и определяется как

$$\rho = \min\left\{\frac{\log|K|}{H(U_i)} : 1 \leq i \leq n\right\},$$

где U_i – случайная величина, принимающая значения из множества возможных значений ключевых материалов, распределяемых i-му участнику, $1 \leq i \leq n$, H – энтропия.

Так как основным назначением схем KPS является минимизация объема памяти, необходимого для хранения ключей, то представляют интерес также следующие параметры, измеряемые числом бит, необходимых для хранения ключей в центре и у каждого участника. Стойкость схемы можно оценивать суммарным количеством неизвестных бит, необходимых для восстановления ключа k_P. Обычно схема считается стойкой, если число бит, которые остаются неизвестными для непривилегированных групп участников, не

[1] Работа выполнена при поддержке гранта Президента РФ НШ № 8564.2006.10.

меньше некоторого заданного числа m. Пусть β_i – суммарная длина ключевых материалов, предоставленных i-му участнику, $1 \le i \le n$, то есть суммарное число бит в их записи (*объем памяти для узла*). *Средний объем памяти схемы* KPS $\overline{\beta}$ – это среднее арифметическое объемов памяти всех узлов схемы. *Общий объем памяти схемы* KPS – это суммарная длина всех различных ключевых элементов, хранящихся в центре.

Например, схема KPS с полной матрицей, в которой каждая пара участников обладает одним общим ключом, причем длины всех ключей одинаковы и равны m, имеет параметры

$$\rho = \frac{1}{n-1}, \quad \overline{\beta} = \beta_i = (n-1)m, \quad \beta = \frac{n(n-1)m}{2}.$$

Данная схема является эталоном, с которым можно сравнивать другие схемы. Ее называют тривиальной, так как ее использование, фактически, равносильно отсутствию какой-либо KPS.

2. Схемы распределения ключей, основанные на схемах пересечений множеств

Определение. Пусть $n > 2$. *Схема распределения ключей на основе шаблонов* KDP(n, k) (*key distribution patterns*) [1], или *схема пересечений множеств* SIS (*set intersection scheme*) [3] в простейшем случае определяется как набор множеств $\{S_1, ..., S_n\}$, являющихся подмножествами множества $\{1, \ldots, k\}$ и удовлетворяющих условию:

(1) если при $i, j, r \in \{1, ..., n\}$ выполнено включение $S_i \cap S_j \subseteq S_r$, то либо $i = r$, либо $j = r$.

По KDP(n, k)-схеме легко построить схему (2,1)-KPS. Для этого надо сформировать множество из k секретных ключей и присвоить им номера 1, …, k. Распределение ключевых материалов осуществляется путем передачи заранее секретным образом каждому абоненту P_i всех ключей с номерами из множества S_i. Теперь для формирования общего ключа участники i и j выбирают ключи, номера которых лежат в пересечении $S_i \cap S_j$, а затем вычисляют общий ключ как сумму или как значение хеш-функции от строки, составленной из этих ключей.

Заметим, что при таком подходе информация о номерах ключей каждого участника не является секретной и может передаваться по сети открытым способом.

Определение [1]. Пусть $w \ge 1$. *Устойчивая к сговору w участников*, или *устойчивая к w-кратной компрометации* схема w-CRKDP(n, k) (*collusion-resistant KDP*) определяется как набор множеств $\{S_1, ..., S_n\}$, являющихся подмножествами множества $\{1, \ldots, k\}$ и удовлетворяющих условию:

(2) если при $i, j, r_1, \ldots, r_w \in \{1, ..., n\}$ выполнено включение $S_i \cap S_j \subseteq \bigcup_{t=1}^{w} S_t$, то либо $i \in \{r_1, \ldots, r_w\}$, либо $j \in \{r_1, \ldots, r_w\}$.

Пусть $g \ge 2$. Определим теперь групповую схему предварительного распределения ключей.

Определение [1]. *w-Устойчивая к сговору схема для групп из g участников* (g, w)-CRKDP(n, k) определяется как набор множеств $\{S_1, ..., S_n\}$, являющихся подмножествами множества $\{1, \ldots, k\}$ и удовлетворяющих условию:

(3) если при $i_1, ..., i_g, p_1, ..., p_w \in \{1, ..., n\}$ выполнено включение $\bigcap_{j=1}^{g} S_{i_j} \subseteq \bigcup_{j=1}^{w} S_{p_j}$, то $\{i_1, ..., i_g\} \cap \{p_1, ..., p_w\} \ne \varnothing$.

3. Связь схем KDP и KPS

Пусть $\boldsymbol{B} = \{B_1, \ldots, B_b\}$ – множество подмножеств множества $\boldsymbol{U}$, называемых блоками, $\boldsymbol{P}, \boldsymbol{F} \subseteq 2^{\boldsymbol{U}}$, $\boldsymbol{P} \cap \boldsymbol{F} = \varnothing$. Схему предварительного распределения ключей на основе пересечений множеств ($\boldsymbol{P}$, $\boldsymbol{F}$)-KDP можно также определить как систему инцидентности ($\boldsymbol{U}$, $\boldsymbol{B}$), удовлетворяющую условию: для всех $F \in \boldsymbol{F}$, таких, что $\boldsymbol{P} \cap F = \varnothing$, выполнено

$$\{B_j : P \subseteq B_j \wedge F \cap B_j = \varnothing\} \ne \varnothing .$$

Последнее условие означает, что для каждой непривилегированной группы участников F найдется блок B_j, включающий всех участников привилегированной группы P, не пересекающийся с F. Данное определение является дуальным к приведенному выше определению KDP(n,q)-схемы. Множество S_j номеров ключей каждого участника в данном случае совпадает с множеством номеров блоков, в которые входит участник j.

Для каждого участника $i \in \boldsymbol{U}$ обозначим $r_i = \left| \{ B_j : i \in B_j \} \right|$, $1 \le i \le n$.

Теорема [6]. Пусть ($\boldsymbol{U}$, $\boldsymbol{B}$) – ($\boldsymbol{P}$, $\boldsymbol{F}$)-KDP и q – степень простого числа. Тогда существует ($\boldsymbol{P}$, $\boldsymbol{F}$)-KPS с ключевым множеством $K = GF(q)$ и информационной скоростью

$$\rho = \frac{1}{\max \{ r_i : 1 \le i \le n \}} .$$

В данном случае ключ k_P для каждой группы P привилегированных участников вычисляется по формуле $k_P = \sum_{i:P\subseteq B_i} s_i$, где $s_i \in GF(q)$ – случайное число, распределяемое каждому участнику, входящему в блок B_i (поэтому каждый участник получает r_i случайных чисел в качестве своих ключевых материалов).

Можно построить (***P***, ***F***)-KPS с более высоким значением информационной скорости. Для каждого $P \in \boldsymbol{P}$ обозначим

$$C_P = \left|\left\{ B_j : P \subseteq B_j \right\}\right| \quad \text{и} \quad D_P = \max\left\{\left|\left\{B_j : P \subseteq B_j \wedge F \cap B_j \neq \varnothing\right\}\right| : F \cap P = \varnothing\right\}.$$

В результате каждый участник группы P имеет C_P общих секретных значений, из которых каждой группе непривилегированных участников известно не более D_P.

Теорема [6]. Пусть (***U***, ***B***) – (***P***, ***F***)-KDP и $m = \min\{ C_P - D_P \}$, $P \in \boldsymbol{P}$. Пусть q – степень простого числа, $q \geq \max\{C_P\} - 1$, $P \in \boldsymbol{P}$. Тогда существует (***P***, ***F***)-KPS с ключевым множеством $K = GF(q)^m$ и информационной скоростью

$$\rho = \frac{m}{\max\{ r_i : 1 \leq i \leq n \}}.$$

Доказательство этой теоремы использует понятие (n, m, t, q)-устойчивой (*resilent*) функции $f : GF(q)^n \to GF(q)^m$, которая остается сбалансированной при фиксации t аргументов, и основано на следующем факте: для любого примарного числа q, $q \geq n - 1$, существует (n, $m - t$, t, q)-устойчивая функция.

4. Построение схем KDP на основе блок-схем

Приведенные выше определения удобно интерпретировать в терминах схем отношений. Напомним, что *конечной структурой инцидентности* (***P***, ***B***, ***I***) называется пара конечных множеств ***P*** и ***B*** (элементы которых называются точками и блоками), связанных некоторым бинарным отношением инцидентности $\boldsymbol{I} \subseteq \boldsymbol{P} \times \boldsymbol{B}$. Определим t-(v, k, λ)-*схему* (или t-*схему*) как структуру инцидентности (***P***, ***B***, ***I***) с $|\boldsymbol{P}| = v$, $|\boldsymbol{B}| = b$, в которой каждый блок имеет мощность k и каждые t точек инцидентны λ блокам [14]. Если при $1 \leq i \leq t$ символом λ_i обозначить число блоков, инцидентных произвольным i точкам, то для таких схем выполняются соотношения

$$bk = vr, \qquad (r = \lambda_1),$$
$$\lambda_i \binom{k-i}{t-i} = \binom{v-i}{t-i}\lambda, \quad 1 \leq i \leq t.$$

В случае схем KDP можно положить $\boldsymbol{P} = \boldsymbol{U} = \{1, \ldots, n\}$ и $\boldsymbol{B} = \{1, \ldots, k\}$ либо рассматривать дуальное представление, рассматривая номера ключей $1, \ldots, k$ как точки, а множества $S_1, \ldots, S_n$ – как блоки. Рассмотрим примеры.

Пример 1. Схема распределения ключей с полной матрицей является KDP(v, b) при $v = n$ и $b = \binom{n}{2} = \frac{n(n-1)}{2}$. Это случай 2-($v$, 2, 1)-схемы, соответствующий тривиальной KPS.

Пример 2. KDP(n, n) получается при выборе в качестве $S_1, \ldots, S_n$ всех подмножеств мощности $n - 1$. Это случай тривиальной 2-(n, $n - 1$, $n - 2$)-схемы.

Более сложный пример можно получить на основе биплоскостей. Напомним, что *биплоскостью* называется симметричная (т. е. $b = v$ и $r = k$) 2-(v, k, 2)-*схема*.

Пример 3 [1]. Каждой биплоскости с $b = v = \frac{1}{2}(r^2 - r + 2)$ блоками соответствует симметричная схема KDP(n, k) с $n = k = \frac{1}{2}(r^2 - r + 2)$, в которой каждое из множеств $S_1, \ldots, S_n$ имеет мощность r. Действительно, каждым двум блокам инцидентны ровно $\lambda = 2$ точки, а каждым двум точкам инцидентны ровно два блока. Поэтому пересечение каждых двух множеств S_i и S_j содержит два номера ключа, причем эта пара номеров ключей лежит только в двух множествах, а именно S_i и S_j.

5. Связь с семействами Шпернера

Множество подмножеств множества $\boldsymbol{U} = \{1, \ldots, n\}$ называется *семейством Шпернера*, если ни одно из них не является подмножеством другого. Легко доказывается следующая

Лемма [1]. Семейство $\{S_1, \ldots, S_n\}$ подмножеств множества $\boldsymbol{U}$, $|\boldsymbol{U}| = k$, образует KDP(n, k)-схему в том и только в том случае, если множество $\{S_i \cap S_j \mid 1 \leq i < j \leq n\}$ образует семейство Шпернера.

Воспользуемся известным результатом Шпернера для доказательства нижней оценки на объем ключевых материалов.

Теорема (*Sperner*, 1928). Если подмножества $S_1, ..., S_n$ множества $\boldsymbol{U}$, $|\boldsymbol{U}| = k$, образуют семейство Шпернера, то $n \le \binom{k}{\lfloor k/2 \rfloor}$. Равенство достигается только в случаях, если $S_1, ..., S_n$ – все m-элементные подмножества множества $\boldsymbol{U}$, где $m = k/2$ при четном k и $m = (k + 1)/2$ или $(k – 1)/2$ при нечетном k.

Следствие 1 [1]. Для любой схемы KDP(n, k) выполняется неравенство $\binom{n}{2} \le \binom{k}{\lfloor k/2 \rfloor}$.

Следствие 2 [1]. Для любой схемы KDP(n, k) каждый абонент должен иметь менее $\log_2 n$ ключей. Если $n \ge 4$, то $k \ge 2 \log_2 n$.

Следствие 3 [1]. Для любой схемы KDP(n, k) и любого i выполняется неравенство $n - 1 \le \binom{r_i}{\lfloor r_i/2 \rfloor}$, где $r_i = |S_i|$, $1 \le i \le n$.

Из этих оценок получаются следующие минимальные значения min b и min r_i для маленьких значений n:

n	3	4	5	6	7	8	9
min b	3	4	5	6	7	7	8
min r_i	2	3	4	4	4	5	5

Еще один тип оценок вытекает из известных комбинаторных соотношений на параметры конфигураций в случае, когда схеме KDP(n, k) соответствует 1-схема, то есть все множества S_i имеют одинаковую мощность, и каждый элемент j, $1 \le j \le n$, входит в одинаковое число r множеств S_i, $1 \le i \le n$. Так, например, из обобщенного неравенства Фишера вытекает оценка $k \ge n$, а также можно показать, что для нетривиальной схемы KDP(n, k) выполняются неравенства $r \ge 3$, $|S_i| \ge 3$ и $|S_i \cap S_j| \ge 2$, $1 \le i, j \le n$.

6. Вероятностный алгоритм построения KDP

Теорема [3]. Существуют KDP(n, k)-схемы с $k \le [13 \log_2 n]$.

Доказательство проводится с помощью вероятностного метода в комбинаторике. В нем множество S_i строится поэлементно в результате реализации последовательности независимых случайных испытаний по схеме Бернулли с вероятностью успеха p. При таком подходе, в общем случае, множества S_i имеют различную мощность. При выборе значения $p = 2/3$ и $k = 13 \log_2 n$ получаем вероятность успеха не менее 1/2.

Если изменить вероятностную схему и потребовать, чтобы множества S_i выбирались случайным образом из множества всех подмножеств фиксированной мощности $|S_i| = t$, то в результате получаются, как правило, более эффективные схемы, содержащие меньшее число ключей. В качестве значения t нужно выбирать то, которое минимизирует вероятность появления «плохих» троек в смысле предыдущей теоремы. В табл. 1 из работы [3] приведены практически вычисленные значения числа ключей для обеих схем.

Таблица 1

Число участников	Число ключей (k)	
(n)	1 способ	2 способ
50	73	48
75	80	53
100	86	56
200	99	64
500	116	75
Число участников	Число ключей (k)	
(n)	1 способ	2 способ
10^3	129	83
10^4	172	111
10^5	215	138
10^6	258	165
10^8	344	220

Поскольку алгоритм построения схемы KDP(n, k) носит вероятностный характер, то после построения требуется проверить, выполняется ли условие (1). Для этого требуется не более $O(n^3k)$ операций. В то же время при очень больших значениях n можно отказаться от этой проверки, так как вероятность построения такой схемы очень мала. Например, при $n = 10^8$ и $k = 399$ она не превышает 10^{-20}. В табл. 2 приведены зна-

чения k, при которых выполняются заданные ограничения на величину β вероятности риска построения первым способом схемы с невыполненным условием (1).

Таблица 2

Число абонентов	Число ключей (k)	
(n)	$\beta \le 10^{-10}$	$\beta \le 10^{-20}$
500	163	254
10^3	171	262
10^4	199	290
10^5	226	317
10^6	253	344
10^8	308	399

Вероятностный алгоритм из предыдущей теоремы можно преобразовать в детерминированный.

Теорема [3]. Схема KDP(n, k) с $k \le 2 \log_2 n$ может быть построена детерминированным алгоритмом за $O(n^3k)$ операций при использовании памяти объема $O(n^3)$.

Как отмечают авторы, полученный детерминированный алгоритм имеет такую же трудоемкость, как и вероятностный, но он не может быть эффективно распараллелен.

7. Схемы KDP, устойчивые к сговору

Рассмотрим теперь примеры схем w-CRKDP(n, k).

Схема из примера 1, очевидно, является w-CRKDP при всех w, $1 \le w \le n$.

Пример 4 [1]. Предположим, имеется 3-(v, k, λ)-схема, параметры которой удовлетворяют неравенству $\lambda_2 > w\lambda_3$ (здесь $\lambda = \lambda_3$). Тогда ей соответствует w-CRKDP(b, v). Действительно, если при $i, j \in \{1, ..., n\}$ и $r_1, ..., r_w \in \{1, ..., n\} \setminus \{i, j\}$ выполнено включение $S_i \cap S_j \subseteq \bigcup_{t=1}^{w} S_t$, то в силу равенств $|S_i \cap S_j| = \lambda_2$ и $|S_i \cap S_j \cap S_{r_t}| = \lambda_3$ при всех $1 \le r_t \le w$ получаем противоречие с неравенством $\lambda_2 > w\lambda_3$.

Лемма [1]. Пусть $w \ge 1$. Семейство $\{S_1, ..., S_n\}$ подмножеств множества $\boldsymbol{U}$, $|\boldsymbol{U}| = k$, образует (w+1)-CRKDP(n, k)-схему в том и только в том случае, если для любой точки j остаточная KDP($n1$, $k - 1$)-схема, полученная из нее удалением одной точки j и оставлением только тех множеств $S_1, ..., S_{n1}$, которые содержат эту точку, является w-CRKDP-схемой.

Так как остаточная схема любой t-схемы является ($t - 1$)-схемой, то из этой леммы вытекает

Следствие [1]. Пусть $w \ge 1$. Любая ($w + 2$)-схема является схемой w-CRKDP.

В качестве общих результатов можно привести следующие оценки.

Теорема [3]. Для любой схемы w-KDP(n, k) выполняется неравенство $k \ge w(\log_2 n - \log_2 w - 1)$.

Теорема [3]. Существует схема w-KDP(n, k), у которой $k \le \lceil ((w+2)^{w+3}/\, 4w^w \log_2 n) \rceil$.

Следствие [3]. Существует схема w-KDP(n, k), имеющая $\lceil 2(w+2)^3 \ln n \rceil$ ключей.

Аналогично рассмотренным выше алгоритмам построения схем KDP может быть предложен алгоритм построения схем w-KDP.

Теорема [3]. Схема w-KDP(n, k) с $k \le \lceil w^3 \ln k \rceil$ может быть построена детерминированным алгоритмом за $O(n^3k)$ операций.

Заметим, что семейства множеств, в которых ни одно из множеств не содержится в объединении w других, подробно изучались в комбинаторике (см., например, работы [11, 13]).

В работе [5] описаны примеры использования циклических геометрий для построения w-KDP-схем. Ниже будут построены w-KDP-схемы, полученные из инверсных плоскостей и плоскостей Минковского. Отметим, что имеются и другие примеры построения w-KDP-схем, например, на основе плоскостей Лагерра (Laguerre) [4].

Напомним, что *инверсная плоскость* (*плоскость Мебиуса*) является 3-($s^2 + 1$, $s + 1$, 1)-схемой с параметрами $v = s^2 + 1$, $k = s + 1$, $b = s(s^2 + 1)$, $r = s(s + 1)$, $\lambda = s + 1$ и представляет собой структуру инцидентности, имеющую точки и блоки, называемые окружностями, такую, что выполняются аксиомы:

C1: любая окружность содержит по крайней мере три точки;

C2: для каждых трех различных точек существует единственная окружность, им инцидентная;

C3: если p и q точки, а C – окружность, содержащая p, но не содержащая q, то существует только одна окружность C', содержащая p и q и имеющая с окружностью C одну общую точку p;

C3: существует четыре точки, не принадлежащие одной окружности.

Пример 5 [5]. Рассмотрим структуру инцидентности ($\boldsymbol{P}$, $\boldsymbol{B}$, $\boldsymbol{I}$) на основе инверсной плоскости порядка s, которая определяется так: множеством точек $\boldsymbol{P}$ (участников) служит множество всех $s^2 + 1$ точек инверсной

плоскости, а множеством блоков ***B*** (ключей) – множество всех $s(s^2+1)$ окружностей. Пусть (i) обозначает множество блоков, инцидентных точке i. Тогда каждой паре абонентов i и j соответствует множество $(i) \cap (j)$, состоящее из окружностей, проходящих через две точки i и j. Если при этом выполнено включение

$$(i) \cap (j) \subseteq \bigcup_{t=1}^{w} (r_t) \ ,$$

то так как через три различные точки проходит не более одной окружности, в объединении справа должно быть, по крайней мере, $|(i) \cap (j)| = s+1$ окружностей. Значит, данная структура образует w-KDP-схему при всех $w \le s$. Для этой схемы $\beta_i = (s^2+1)\,l$, где длину ключевого материала l можно положить равной $\left\lceil \frac{m}{s-w+1} \right\rceil$. В этом случае будет выполнено условие стойкости w-KDP-схемы, которое означает, что после того, как w участников $\{r_1, \ldots, r_w\}$ объединят свои ключевые материалы с целью восстановления общего ключа участников i и j, неизвестными останутся по крайней мере m бит. Средний и общий объем памяти схемы будут равны соответственно

$$\overline{\beta} = (s^2+1)\left\lceil \frac{m}{s-w+1} \right\rceil \qquad \text{и} \qquad \beta = \sum_{x \in B} l(x) = (s^3+s)\left\lceil \frac{m}{s-w+1} \right\rceil .$$

Пример 6 [5]. Рассмотрим остаточную структуру инцидентности (***P′***, ***B′***, ***I***), полученную из инверсной плоскости порядка s удалением одной точки и всех окружностей, содержащих эту точку. Тогда, очевидно, она также будет w-KDP-схемой при всех $w \le s-1$. Для этой схемы $\beta_i = (s^2-1)\,l$, где в качестве длины ключевого материала l можно выбрать величину $\lceil m / (s-w) \rceil$. В этом случае также будет выполнено условие стойкости w-KDP-схемы. Средний и общий объем памяти схемы будут равны соответственно

$$\overline{\beta} = (s^2-1)\left\lceil \frac{m}{s-w} \right\rceil \qquad \text{и} \qquad \beta = \sum_{i=1}^{|B|} \beta_i = (s^3-s^2)\left\lceil \frac{m}{s-w} \right\rceil ,$$

так как число слагаемых в последней сумме равно $(s^3+s)-(s^2+s) = s^3-s^2$.

Конечной плоскостью Минковского ([14]) называется структура инцидентности с точками, прямыми и окружностями, в которой множество всех прямых разбивается на два класса L_1 и L_2 так, что выполняются аксиомы:

M1: каждая точка лежит на единственной прямой из каждого класса L_1 и L_2, каждая прямая класса L_1 пересекается с каждой прямой класса L_2 в единственной точке и прямые и окружности пересекаются в единственной общей точке;

M2: любые три точки, из которых никакие две не коллинеарны, инцидентны одной единственной окружности;

M3: если p и q – неколлинеарные точки и если C – окружность, содержащая p, но не содержащая q, то существует только одна окружность C', содержащая p и q и имеющая с окружностью C одну общую точку p;

M4: существует окружность, содержащая по крайней мере три точки.

Для каждой плоскости Минковского существует целое число s, которое называется порядком плоскости (в этом случае саму плоскость обозначают через $M(s)$), такое, что:

1) каждая окружность имеет $(s+1)$ точку;

2) плоскость $M(s)$ имеет $(s+1)^2$ точек, $(s+1)$ прямую в каждом из двух классов и $s(s^2-1)$ окружностей;

3) каждая прямая имеет $(s+1)$ точку;

4) каждая точка лежит на двух прямых и $s(s-1)$ окружностях;

5) каждые две неколлинеарные точки лежат на $(s-1)$ окружностях.

Пример 7 [5]. Пусть $2 \le u \le s$. Рассмотрим структуру инцидентности (***P***, ***B***, ***I***) на основе инверсной плоскости Минковского порядка s. Для этого зафиксируем u точек $R_1, \ldots, R_u$, лежащих на одной прямой. Пусть множеством точек P (абонентов) служит множество всех точек плоскости Минковского, не лежащих на прямой, содержащих данные точки, а множеством блоков B (ключей) – множество всех окружностей, проходящих через точки $\{r_1, \ldots, r_u\}$, и некоторую пару точек i и j, лежащих на одной прямой. Тогда паре абонентов i и j соответствует множество окружностей, проходящих через точки i, j и $r_1, \ldots, r_u$. Данная структура имеет s^2+s точек и $u(s^3+s)+(s^2+s)(2s-1)/2$ блоков (ключей). При этом имеется us точек, инцидентных $(u-1)(s-1)+2s-1$ ключам, и s^2+s-us точек, инцидентных $u(s-1)+2s-1$ ключам. Полагаем длину ключевого материала равной $l = \lceil m / (u-w-2) \rceil$, если блок является окружностью, и m в противном случае. При таком выборе будет выполнено условие стойкости w-KDP-схемы, и она будет иметь следующие параметры:

$$\bar{\beta} = \frac{u(m-1)s(s-1)+(s^2+s+us)u(s-1)}{s^2+s}\left\lceil \frac{m}{u-w-2} \right\rceil + (2s-1)m\,,$$

$$\beta = u(s^2-s)\left\lceil \frac{m}{u-w} \right\rceil + \frac{(s^2+s)(2s-1)m}{2}\,.$$

8. Групповые схемы KDP, устойчивые к сговору

Для групповых схем известны следующие примеры.

Лемма [1]. Пусть $g \geq 1$, $w \geq 1$. Семейство $\{S_1, ..., S_n\}$ подмножеств множества $\boldsymbol{U}$, $|\boldsymbol{U}| = k$, образует (g, w+1)-CRKDP(n, k)-схему в том и только в том случае, если для любой точки j остаточная KDP(n_1, $k-1$)-схема, полученная из нее удалением одной точки j и оставлением только тех множеств S_1, ..., S_{n1}, которые содержат эту точку, является (g, w)-CRKDP-схемой.

Следствие [1]. Пусть $g \geq 1$, $w \geq 1$. Любая ($g + w$)-схема является схемой (g, w)-CRKDP.

Теорема [1]. Число ключей любой (g, w)-KDP(n, k)-схемы удовлетворяет неравенству

$$k \geq w(g \log_2 n - \log_2 w - g \log_2 g).$$

Теорема [1]. Существует (g, w)-KDP(n, k)-схема, у которой $k \leq \lceil ((w+g)^{w+g+1}/\, g^g w^w \log_2 n\,) \rceil$.

Заметим, что изучаемый в комбинаторике тип (i, j)-семейств без перекрытий, по сути, совпадает с понятием схемы (g, w)-KDP.

Определение. Если $\boldsymbol{X}$ – множество из v элементов, $|\boldsymbol{X}| = v$, а $\boldsymbol{F}$ – множество его подмножеств (блоков), $|\boldsymbol{F}| = b$, то ($\boldsymbol{X}$, $\boldsymbol{F}$) называется (i, j)-*семейством без перекрытий* (*cover-free family*) и обозначается (i, j)-CFF(v, b), если для любых i блоков $B_1, ..., B_i \in \boldsymbol{F}$ и любых j блоков $A_1, ..., A_j \in \boldsymbol{F}$ выполняется условие

$$\bigcap_{k=1}^{i} B_k \not\subset \bigcup_{s=1}^{j} A_s\,.$$

Такие семейства были введены в рамках теории кодирования в [9] (см. также [10]).

В работе [8] для построения семейств (i, j)-CFF(v, b) используются системы (i, j)-SS(v,b).

Определение [13]. (i, j)-*разделенная система* (*separating system*) – это пара множеств ($\boldsymbol{X}$, $\boldsymbol{B}$), такая, что для всех P, $Q \subseteq \boldsymbol{X}$ с условием $|P| \leq i$, $|Q| \leq j$ и $P \cap Q = \varnothing$, существует блок $B \in \boldsymbol{B}$, такой, что либо $P \subseteq B$ *и* $Q \cap B = \varnothing$, либо $Q \subseteq B$ *и* $P \cap B = \varnothing$. Если $|X| = v$ и $|\boldsymbol{B}| = b$, то используется обозначение (i, j)-SS(v, b).

Определение [8]. (N, n, m, $\{i, j\}$)-*разделенное семейство хеш-функций* (*separating hash family*) (обозначается как SHF(N; n, m, $\{i, j\}$)) – это множество из N функций f: $Y \to X$, $|Y| = n$, $|X| = m$, такое, что для всех f и для любых подмножеств C_1, $C_2 \subseteq \{1, 2, \ldots, n\}$, $|C_1| = i$, $|C_2| = j$ и $C_1 \cap C_2 = \varnothing$, существует функция f, такая, что

$$\{f(y):\ y \in C_1\} \cap \{f(y):\ y \in C_2\} = \varnothing.$$

Лемма [7]. Каждая схема (i, j)-SS(v, b) эквивалентна семейству SHF(b; v, 2, $\{i, j\}$).

Теорема [7]. Если существует система (i, j)-SS(v, b), то существует (i, j)-CFF(2v, b).

9. Построение больших схем KDP из малых

В работе [1] указано три способа построения новых KDP(n, k) на основе уже построенных ранее. Не углубляясь в детали, отметим только, что по двум схемам KDP(n_1, k_1) и KDP(n_2, k_2) можно построить схему KDP(n, k) с параметрами:

(a) $n = n_1 \times n_2$, $k = k_1 + k_2$;

(b) $n = n_1 + r_1(n_2 - 1)$, $k = k_1 + k_2 - 1$;

(c) $n = n_1 + n_2 + (r_1 - 1)(r_2 - 1) - 1$, $k = k_1 + k_2 - 2$,

где r_1 и r_2 – число множеств первой и второй схем соответственно, содержащих произвольную точку (номер ключа). Например, начиная с двух KDP(7, 7), соответствующих двум биплоскостям, каждая из которых является 2-(7, 4, 2)-схемой, с применением этих конструкций можно соответственно построить:

(a) KDP(49, 14), в которой множества S_i имеют мощность 8;

(b) KDP(31, 13), в которой множества S_i имеют мощность 9;

(c) KDP(22, 12), в которой множества S_i имеют мощность 10.

Утверждение [1]. Если схемы KDP(n_1, k_1) и KDP(n_2, k_2) являются w-CRKDP-схемами, то схема KDP(n, k), построенная из них с помощью конструкций (a), (b) и (c), также является w-CRKDP-схемой.

В заключение приведем еще один простой и эффективный индуктивный способ построения KDP(n, k)-схем, предложенный Д.Б. Тишуриным и Р.В. Шпаком. Определим *матрицу инцидентности* KDP(n, k)-схемы

как (0, 1)-матрицу размера $n \times k$, в которой столбцы соответствуют элементам множества S, а строки – подмножествам S_1, ..., S_n, причем единицы стоят на пересечении со столбцами, помеченными элементами подмножества, соответствующего строке. Например, KDP(n, n)-схеме из примера 2 соответствует матрица инцидентности

$$\begin{pmatrix} 11...10 \\ 11...01 \\ ... \\ 01...11 \end{pmatrix}.$$

Заметим, что любая перестановка строк и столбцов матрицы инцидентности KDP(n, q)-схемы преобразует ее в матрицу KDP(n, k)-схемы.

Лемма. (0, 1)-матрица размера $n \times q$ задает схему KDP(n, k) в том и только в том случае, если для любых трех ее строк найдутся три столбца, на пересечении которых стоит матрица инцидентности схемы KDP(3, 3).

Доказательство. Пусть строкам матрицы соответствуют множества S_1, ..., S_n. Тогда утверждение леммы вытекает из следующего очевидного замечания: для любых трех строк i, j, r существование столбца, на пересечении с которым в этих строках стоят элементы 1, 1 и 0 соответственно эквивалентно условию $S_i \cap S_j \not\subset S_r$.

Теорема. Если существует KDP(n, k)-схема, то существует и KDP(n^2, $3q$)-схема.

Доказательство. Пусть имеется KDP(n, k)-схема с матрицей инцидентности A. Для $1 \le i \le n$ определим матрицы A_i и B_i, первая из которых получается из A циклической перестановкой строк вверх на $i - 1$ шагов, а вторая составлена из одинаковых строк, каждая из которых совпадает с i-й строкой матрицы A. Рассмотрим матрицу

$$\left(\begin{array}{ccc} A & A & B \\ A & A_2 & B_2 \\ ... & ... & ... \\ A & A_n & B_n \end{array}\right) = \begin{pmatrix} C_1 \\ C_2 \\ ... \\ C_n \end{pmatrix}.$$

Покажем, что эта матрица является матрицей инцидентности KDP(n^2, $3q$)-схемы. В силу леммы достаточно показать, что для произвольных трех строк этой матрицы найдутся три столбца, на пересечении которых стоит матрица инцидентности KDP(3, 3)-схемы.

Если три строки лежат в одной подматрице $C_i = (A, A_i, B_i)$, то такие столбцы найдутся в ее первой подматрице, совпадающей с матрицей A. Если три строки лежат в разных подматрицах C_i, C_j, C_k, то такие столбцы найдутся в подматрицах B_i, B_j, B_k.

Пусть теперь три строки лежат в двух подматрицах $C_i = (A, A_i, B_i)$ и $C_j = (A, A_i, B_j)$. Пусть две строки лежат в подматрице C_i, а одна в подматрице C_j. Покажем, что найдутся три столбца, на пересечении которых стоит одна из матриц вида

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \quad \text{или} \quad \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

Для удобства будем называть матрицы A, A_i и B_i первой, второй и третьей колонками матрицы C_i. Если в первой колонке все три строки разные, то такие столбцы найдутся уже в первой колонке. Пусть теперь среди двух строк первой колонки две совпадают. Аналогично, если во второй колонке строки различны, то такие столбцы найдутся во второй колонке. Поэтому предположим, что во второй колонке также две строки совпадают. При этом совпадать может только третья строка с одной из двух первых, причем в первой и второй колонках совпадают разные строки. В третьей колонке всегда совпадают первые две строки, так как они берутся из одной матрицы B_i.

Осталось заметить, что в силу основного свойства KDP-схемы в ее матрице инцидентности для любых трех строк, две из которых совпадают, всегда найдется столбец, на пересечении которого с совпадающими строками стоят единицы, а на пересечении с третьей строкой – ноль.

Следствие. Если существует KDP(n_0, k_0)-схема, то для любого натурального t существует KDP(n, k)-схема с параметрами $n = {n_0}^{2^t}$, $q = 3^t k_0$ и $k = k_0 \ (\log_{n_0} n)^{\log_2 3}$.

Доказательство получается применением теоремы t раз. В результате строится схема KDP(n,q) с $n = {n_0}^{2^t}$ и $k = 3^t k_0$. Выражая из этих равенств t и приравнивая выражения, получаем равенство $\ln k = \log_2 3 \ln \log_2 (\log_{n_0} n) + \ln k_0$, откуда и вытекает требуемое равенство.

Например, начиная со схем KDP(3, 3), KDP(4, 4) и KDP(5, 5), можно построить схемы со следующими параметрами:

n	k
81	27
256	36
625	45
6561	81
65536	108
390625	135
43046721	243
4294967296	324
152587890625	405

Оценим объем ключевых материалов, получаемых каждым участником. Если в исходной KDP(n, k)-схеме все множества S_1, ..., S_n имеют одинаковое число элементов, равное m, $2 < m < k$, то для схемы KDP(n^2, $3k$) имеем $\overline{\beta} = \beta_i = 3m, \quad \beta = 3k$.

Заметим также, что если к матрице инцидентности KDP(n^2, $3k$)-схемы, построенной из KDP(n, k)-схемы с использованием предыдущей теоремы, добавить три строки вида (0…01...11...1), (1...10...01...1) и (1...11...10...0), то получится KDP(n^2+3, $3k$)-схема. С использованием этого факта можно построить, например, KDP(n, k)-схемы с параметрами

n	k
147	27
364	36
787	45
21612	81
132499	108
619372	135
467078547	243
17555985004	324
383621674387	405

ЛИТЕРАТУРА

1. *Mitchell C.J.*, *Piper C.* Key storage in Secure Networks // Discrete and Applied Math. 1988. 21. P. 215 – 228.
2. *Mitchell C.J.* Combinatorial techniques for key storage reduction in secure networks // Technical memo. Hewlett-Packard Laboratories. Bristol, 1988.
3. *Dyer M.*, *Fenner T.*, *Frieze A.*, *Thomason A.* On key storage in secure networks // J. Cryptology. 1995. V. 8. P. 189 – 200.
4. *O'Keefe C.M.* Applications за finite geometries in information security // Australas. J. Combin. 1993. V. 7. P. 195 – 212.
5. *O'Keefe C.M.* Key distribution patterns using Minkowski planes // Design, Codes and Cryptography. 1995. V. 5. No. 3. P. 261 – 267.
6. *Stinson D.R.*, *van Trung T.* Some new results on key distribution patterns and broadcast encryption // Designs, Codes and Cryptography. 1998. V. 14. P. 261 – 279.
7. *Stinson D.R.*, *van Trung T*, *Wei R.* Secure frameproof codes, key distribution patterns, group testing algorithms and related structures // J. Statist. Plan. Infer. May 2000. V. 86. No. 2. P. 595 – 617.
8. *Stinson D.R.*, *Wei R.*, *Zhu L.* New constructions for perfect hash families and related structures using combinatorial designs and codes // J. Combinatorial Designs. 2000.V. 8. No. 3. P. 189 – 200.
9. *Kautz W.H.*, *Singleton R.S.* Nonrandom binary superimposed codes // IEEE Trans. Inform. Theory. 1964. V. 10. P. 363 – 377.
10. *Dyachkov A.G.*, *Rykov V.V.*, *Rashad A.M.* Superimposed distance codes // Problems of Control and Information Theory. 1989. V. 18. P. 237 – 250.
11. *Erdösh P.*, *Frankl P.*, *Füredi Z.* Families of finite sets in which no set is covered by the union of two others // J. Combinatorial Theory. 1982. V. A33. P. 158 – 166.
12. *Erdösh P.*, *Frankl P.*, *Füredi Z.* Families of finite sets in which no set is covered by the union of r others // Israel J. Mathematics. 1985. V. 51. P. 75 – 89.
13. *Friedeman A.D.*, *Graham R.L.*, *Ullman J.D.* Universal single transition time asynchronous state assignments // IEEE Trans. Comput. 1969. V. 18. P. 541 – 547.
14. *Dembovski P.* Finite geometries. Berlin; Heidelberg: Springer Verlag, 1968.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

DOI 10.17223/20710410/1/11

УДК 004.94

БАЗОВАЯ РОЛЕВАЯ ДП-МОДЕЛЬ

П.Н. Девянин

Институт криптографии, связи и информатики Академии ФСБ России, г. Москва

E-mail: peter_devyanin@hotmail.com

На основе ролевых моделей (*RBAC*) и ДП-моделей компьютерных систем с дискреционным или мандатным управлением доступом строится базовая ролевая ДП-модель. С учетом фактических ролей, прав доступа и возможных действий недоверенных сессий анализируются правила преобразования состояний системы. Обосновываются условия передачи прав доступа ролей сессиями пользователей.

Ключевые слова: *компьютерная безопасность*, *математические модели безопасности*, *ролевая модель.*

Ролевое управление доступом (РУД) [1 – 3] является современным, эффективным механизмом защиты компьютерных систем (КС). С использованием иерархии ролей возможно обеспечение управления доступом, точно соответствующего должностным обязанностям пользователей КС. При этом используемые в РУД механизмы статических и динамических ограничений позволяют реализовать на основе РУД дискреционное или мандатное управление доступом.

Модифицируем определения семейства ролевых моделей *RBAC* [1 – 3], базовой ДП-модели, БК ДП-модели, ФАС ДП-модели и мандатной ДП-модели [4] для обеспечения возможности анализировать условия реализации информационных потоков по памяти и по времени в КС с РУД. Модифицированную ДП-модель будем называть базовой ролевой ДП-моделью (или, сокращенно, БР ДП-моделью). При этом используем следующие обозначения:

$E = O \cup C$ – множество сущностей, где O – множество объектов, C – множество контейнеров;

U – множество пользователей, при этом пользователи по определению не являются сущностями;

L_U – множество доверенных пользователей;

N_U – множество недоверенных пользователей, при этом выполняются равенства: $L_U \cup N_U = U$, $L_U \cap N_U = \varnothing$;

$S \subseteq E$ – множество субъект-сессий пользователей;

L_S – множество доверенных субъект-сессий;

N_S – множество недоверенных субъект-сессий, при этом выполняется равенство $L_S \cap N_S = \varnothing$;

R – множество ролей;

AR – множество административных ролей ($AR \cap R = \varnothing$);

$R_r = \{read_r, write_r, append_r, execute_r, own_r\}$ – множество видов прав доступа;

$R_a = \{read_a, write_a, append_a, own_a\}$ – множество видов доступа;

$R_f = \{write_m, write_t\}$ – множество видов информационных потоков;

$R_{raf} = R_r \cup R_a \cup R_f$ – множество видов прав доступа, видов доступа и видов информационных потоков, при этом множества R_r, R_a, R_f попарно не пересекаются;

$P \subset E \times R_r$ – множество прав доступа к сущностям;

UA: $U \to 2^R$ – функция авторизованных ролей пользователей;

AUA: $U \to 2^{AR}$ – функция авторизованных административных ролей пользователей;

PA: $R \to 2^P$ – функция прав доступа ролей;

user: $S \to U$ – функция принадлежности субъект-сессии пользователю, задающая для каждого субъект-сессии пользователя, от имени которого она активизирована;

roles: $S \to 2^R \cup 2^{AR}$ – функция текущих ролей субъект-сессий, задающая для пользователя роли, на которые авторизован активизированный от его имени данный субъект-сессия, при этом в каждом состоянии системы для каждого субъект-сессии $s \in S$ выполняется включение $roles(s) \subseteq UA(user(s)) \cup AUA(user(s))$;

can_manage_rights: $AR \to 2^R$ – функция администрирования прав доступа ролей, определяющая для каждой административной роли множество ролей, для которых разрешено включать или удалять права доступа во множества их прав доступа с использованием данной административной роли.

Определение 1. Иерархией сущностей называется заданное на множестве сущностей E отношение частичного порядка «≤», удовлетворяющее условию: если для сущности $e \in E$ имеются сущности $e_1, e_2 \in E$, такие, что $e \leq e_2$, $e \leq e_1$, то $e_1 \leq e_2$ или $e_2 \leq e_1$. В случае, когда для двух сущностей $e_1, e_2 \in E$ выполняются условия $e_1 \leq e_2$ и $e_1 \neq e_2$, будем говорить, что сущность e_1 содержится в сущности-контейнере e_2, и будем использовать обозначение $e_1 < e_2$.

Определение 2. Определим H_E: $E \to 2^E$ – функцию иерархии сущностей, сопоставляющую каждой сущности $c \in E$ множество сущностей $H_E(c) \subset E$ и удовлетворяющую следующим условиям.

Условие 1. Если сущность $e \in H_E(c)$, то $e < c$ и не найдется сущности-контейнера $d \in C$, такой, что $e < d$, $d < c$.

Условие 2. Для любых сущностей $e_1, e_2 \in E$, $e_1 \neq e_2$, по определению выполняются равенство $H_E(e_1) \cap H_E(e_2) = \varnothing$ и условия:

- если $o \in O$, то выполняется равенство $H_E(o) = \varnothing$;
- если $e_1 < e_2$, то или $e_1, e_2 \in E \setminus S$, или $e_1, e_2 \in S$;
- если $e \in E \setminus S$, то $H_E(e) \subset E \setminus S$;
- если $s \in S$, то $H_E(s) \subset S$.

Определение 3. Иерархией ролей в БР ДП-модели называется заданное на множестве ролей R отношение частичного порядка «≤». При этом по определению выполняется условие: для пользователя $u \in U$, если роли $r, r' \in R$, такие, что $r \in UA(u)$ и $r' \leq r$, то выполняются условия $r' \in UA(u)$. В случае, когда для двух ролей $r_1, r_2 \in R$ выполняются условия $r_1 \leq r_2$ и $r_1 \neq r_2$, будем использовать обозначение $r_1 < r_2$.

Определение 4. Определим H_R: $R \to 2^R$ – функцию иерархии ролей, сопоставляющую каждой роли $r \in R$ множество ролей $H_R(r) \subset R$ и удовлетворяющую условию: если роль $r' \in H_R(r)$, то $r' < r$ и не существует роли $r'' \in R$, такой, что $r' < r''$, $r'' < r$.

Определение 5. Иерархией административных ролей в БР ДП-модели называется заданное на множестве ролей AR отношение частичного порядка «≤». При этом по определению выполняется условие: для пользователя $u \in U$, если административные роли $r, r' \in AR$, такие, что $r \in AUA(u)$ и $r' \leq r$, то выполняются условия $r' \in AUA(u)$. В случае, когда для двух ролей $r_1, r_2 \in AR$ выполняются условия $r_1 \leq r_2$ и $r_1 \neq r_2$, будем использовать обозначение $r_1 < r_2$.

Определение 6. Определим H_{AR}: $AR \to 2^{AR}$ – функцию иерархии административных ролей, сопоставляющую каждой роли $r \in AR$ множество ролей $H_{AR}(r) \subset AR$ и удовлетворяющую условию: если роль $r' \in H_{AR}(r)$, то $r' < r$ и не существует роли $r'' \in AR$, такой, что $r' < r''$, $r'' < r$.

Определение 7. Пусть определены: множества пользователей U, сущностей E, субъект-сессий S, прав доступа к сущностям P, доверенных пользователей L_U, доверенных субъект-сессий L_S, множество доступов субъект-сессий к сущностям $A \subseteq S \times E \times R_a$, множество информационных потоков $F \subseteq E \times E \times R_f$, функции авторизованных ролей пользователей UA, авторизованных административных ролей пользователей AUA, прав доступа ролей PA, принадлежности субъект-сессии пользователю *user*, текущих ролей субъект-сессии *roles*, иерархии ролей H_R, иерархии административных ролей H_{AR}, иерархии сущностей H_E. Определим $G = (UA, AUA, PA, user, roles, A, F, H_R, H_{AR}, H_E, L_U, L_S)$ – состояние системы.

Используем обозначения:

$\Sigma(G^*, OP)$ – система, при этом: G^* – множество всех возможных состояний, OP – множество правил преобразования состояний G Ã $_{op}$ G' – переход системы $\Sigma(G^*, OP)$ из состояния G в состояние G' с использованием правила преобразования состояний $op \in OP$.

Если для системы $\Sigma(G^*, OP)$ определено начальное состояние, то будем использовать обозначение: $\Sigma(G^*, OP, G_0)$ – система $\Sigma(G^*, OP)$ с начальным состоянием G_0.

БР ДП-модель предназначена для анализа условий реализации в КС с РУД информационных потоков, и в ее рамках не предполагается исследовать вопросы администрирования множества ролей, иерархии ролей, иерархии административных ролей, множеств авторизованных ролей пользователей, параметров ограничений. Таким образом, будем использовать следующее предположение.

Предположение 1. В рамках БР ДП-модели на траекториях функционирования системы не изменяются: значения множеств U, L_U, R, функции UA, AUA, H_R, H_{AR}. В БР ДП-модели не используются динамические ограничения.

В БР ДП-модели пользователи или субъект-сессии могут быть доверенными или недоверенными. При этом в отличие от доверенных субъектов систем с дискреционным управлением доступом доверенные пользователи или субъект-сессии могут не обладать ролями, включающими все права доступа ко всем сущностям системы. Чтобы не усложнять описание правил преобразования состояний системы в рамках БР ДП-модели, целесообразно использовать следующие предположение и определение.

Предположение 2. Каждый пользователь или субъект-сессия системы $\Sigma(G^*, OP)$ вне зависимости от имеющихся у нее авторизованных ролей является либо доверенной, либо недоверенной. Доверенные поль-

зователи или субъект-сессии не создают новых субъект-сессий. Каждый недоверенный пользователь или субъект-сессия могут создать только недоверенную субъект-сессию.

Определение 8. Доверенную субъект-сессию назовем корректной относительно информационных потоков по времени, если она не участвует в их реализации.

Используем обозначения:

$LF_S \subset L_S$ – множество доверенных субъект-сессий корректных относительно информационных потоков по времени, $NF_S \subset L_S$ – множество доверенных субъект-сессий некорректных относительно информационных потоков по времени, при этом по определению выполняются равенства $LF_S \cap NF_S = \varnothing$, $LF_S \cup NF_S = L_S$.

В рамках предположений 1 и 2 будем использовать следующее сокращенное обозначение для состояния системы $G = (PA, user, roles, A, F, H_E)$.

Предположение 3. Только информационный поток по памяти к сущности, функционально ассоциированной с субъект-сессией, приводит к изменению вида преобразования данных, реализуемого этой субъект-сессией. Функционально ассоциированными с субъект-сессией являются сущности, от которых зависит вид преобразования данных, реализуемого субъект-сессией в данном или некотором последующем состоянии системы $\Sigma(G^*, OP)$. Множество сущностей, функционально ассоциированных с субъект-сессией, не изменяется в процессе функционирования системы.

В существующих КС при создании субъект-сессии множество функционально-ассоциированных с ней сущностей может задаваться в зависимости от многих параметров (например, в зависимости от пользователя, создающего субъект-сессию, от сущности, из которой создается субъект-сессия, или от компьютера, на котором создается субъект-сессия). Для упрощения описания правил преобразования состояний в дальнейшем будем использовать следующее предположение.

Предположение 4. При создании новой субъект-сессии s множество функционально ассоциированных с ней сущностей задается только в зависимости от сущности, из которой создается субъект-сессия s, и пользователя, который либо создает субъект-сессию s, либо от имени которого другая субъект-сессия создает субъект-сессию s.

Используем следующие обозначения:

$[s] \subset E \cup U$ – множество сущностей, функционально ассоциированных с субъект-сессией s (при этом по определению выполняется условие $s \in [s]$), и пользователей, каждый из которых может создать субъект-сессию, являющуюся функционально ассоциированной сущностью с субъект-сессией s.

fa: $U \times E \to 2^E \cup 2^U$ – функция, задающая множества сущностей, функционально ассоциированных с субъект-сессией, при ее создании пользователем (или от имени пользователя другой субъект-сессией) из сущности.

Определение 9. Доверенную субъект-сессию y назовем функционально корректной, если во множество функционально ассоциированных с ней сущностей $[y]$ не входят недоверенные субъект-сессии.

Определение 10. Доверенную субъект-сессию y назовем корректной относительно доверенной субъект-сессии y' и сущности e (не являющейся доверенной субъект-сессией), если субъект-сессия y не реализует информационный поток по памяти от сущности e к сущности e', функционально ассоциированной с доверенной субъект-сессией y'.

Используем обозначение:

$y(E) \subset E \times L_S$ – множество пар вида доверенная субъект-сессия и сущность, относительно которых корректна доверенная субъект-сессия y.

Предположение 5. Субъект-сессии могут иметь друг к другу только доступ владения own_a. Роли могут обладать к субъект-сессиям только правом доступа владения own_r. Если субъект-сессия s_1 реализовала информационный поток по памяти от себя к сущности, функционально ассоциированной с другой субъект-сессией s_2, то субъект-сессия s_1 получает: доступ владения own_a к субъект-сессии s_2, возможность использовать роли из множества ролей $roles(s_2)$ (при этом субъект-сессия s_1 не может изменить множество текущих ролей $roles(s_2)$), возможность получать доступ владения own_a к субъект-сессиям, доступом владения к которым обладает субъект-сессия s_2, возможность использовать административные роли субъект-сессии s_2 для осуществления действий над ролями и сущностями, которые позволяют ей выполнять права доступа ролей субъект-сессии s_2.

Используем следующие обозначения:

de_facto_roles: $S \to 2^{R \cup AR}$ – функция фактических текущих ролей субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s_1 \in S$ выполняется равенство: $de\_facto\_roles(s_1) = roles(s_1) \cup \{r \in R \cup AR$: существует $s_2 \in S$, $(s_1, s_2, own_a) \in A$ и $r \in roles(s_2)\}$.

de_facto_rights: $S \to 2^P$ – функция фактических текущих прав доступа субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s \in S$ выполняется равенство: $de\_facto\_rights(s) = \{p \in P$: существует $r \in de\_facto\_roles(s)$ и $p \in PA(r)\}$.

de_facto_actions: $S \to 2^P \times 2^R$ – функция фактических возможных действий субъект-сессий, при этом по определению в каждом состоянии системы $G = (PA, user, roles, A, F, H_E)$ для каждой субъект-сессии $s_1 \in S$ выполняется равенство: $de\_facto\_actions(s_1) = (PA(roles(s_1)) \times can\_manage\_rights(roles(s_1) \cap AR)) \cup \{(p, r) \in P \times R$: существует $s_2 \in S$, $(s_1, s_2, own_a) \in A$, $r \in can\_manage\_rights(AR \cap roles(s_2))$ и $p \in PA(roles(s_2))\}$.

В рамках БР ДП-модели используются следующие 19 правил преобразования состояний: *take_role*(x, r), *remove_role*(x, r), *grant_right*$(x, r, (y, \alpha_r))$, *remove_right*$(x, r, (y, \alpha_r))$, *create_entity*(x, r, y, z), *create_first_session*(u, r, y, z), *create_session*(x, r, y, z), *delete_entity*(x, y, z), *rename_entity*(x, y, z), *control*(x, y, z), *access_own*(x, y), *take_access_own*(x, y, z), *access_read*(x, y), *access_write*(x, y), *access_append*(x, y), *flow*(x, y, y', z), *find*(x, y, z), *post*(x, y, z), *pass*(x, y, z). Дадим определение.

Определение 11. Монотонное правило преобразования состояний – правило преобразования состояний из множества *OP*, применение которого не приводит к удалению из состояний: ролей из множества текущих ролей субъект-сессии, прав доступа ролей к сущностям, субъект-сессий или сущностей, доступов субъект-сессий к сущностям, информационных потоков.

По определению 11 немонотонными правилами преобразования состояний будут являться: *remove_role*(x, r), *remove_right*$(x, r, (y, \alpha_r))$, *delete_entity*(x, y, z). Возможно доказать утверждение 1.

Утверждение 1. Пусть $G_0 = (PA_0, user_0, roles_0, A_0, F_0, H_{E0})$ – начальное состояние системы $\Sigma(G^*, OP, G_0)$. Пусть также существуют состояния системы $G_1, \ldots, G_N = (PA_N, user_N, roles_N, A_N, F_N, H_{EN})$ и правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G_0\ \tilde{A}_{op1}\ G_1\ \tilde{A}_{op2} \ldots \tilde{A}_{opN}\ G_N$, где $N \geq 0$. Тогда существуют состояния $G'_1, \ldots, G'_M = (PA'_M, user'_M, roles'_M, A'_M, F'_M, H'_{EM})$, где $M \geq 0$, и монотонные правила преобразования состояний $op'_1, \ldots, op'_M$ такие, что $G_0\ \tilde{A}_{op'1}\ G'_1\ \tilde{A}_{op'2} \ldots \tilde{A}_{op'M}\ G'_M$ и выполняются следующие условия.

Условие 1. Верно включение $S_N \subset S'_M$, и для каждого субъект-сессии $s \in S_N$ выполняются условия: $user_N(s) = user'_M(s)$, $roles_N(s) \subset roles'_M(s)$.

Условие 2. Верно включение $E_N \subset E'_M$, и для каждой сущности $e \in E_N$ выполняется условие $H_{EN}(e) \subset H'_{EM}(e)$.

Условие 3. Для каждой роли $r \in R$ выполняется условие $PA_N(r) \subset PA'_M(r)$.

Условие 4. Верно включение $A_N \subset A'_M$.

Условие 5. Верно включение $F_N \subset F'_M$.

Таким образом, в рамках БР ДП-модели при анализе условий передачи прав доступа, реализации информационных потоков по памяти или по времени возможно использование только монотонных правил преобразования состояний. При исследовании в статье условий передачи прав доступа ролей кооперирующими субъект-сессиями пользователей не применяются следующие монотонные правила: *create_entity*(x, r, y, z), *create_session*(x, r, y, z), *rename_entity*(x, y, z), *access_read*(x, y), *flow*(x, y, y', z), *find*(x, y, z), *pass*(x, y, z). Данные правила преобразования состояний введены в БР ДП-модель для анализа условий реализации информационных потоков по времени. Следовательно, приведем в таблице условия и результаты применения только правил преобразования состояний, которые необходимы для анализа условий передачи прав доступа ролей.

Монотонные правила преобразования состояний, используемые для передачи прав доступа ролей

Правило	Исходное состояние $G = (PA, user, roles, A, F, H_E)$	Результирующее состояние $G' = (PA', user', roles', A', F', H_E')$
1	2	3
take_role(x, r)	$x \in S$, $r \in UA(user(x)) \cup AUA(user(x))$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $A' = A$, $F' = F$, $H_E' = H_E$, $roles'(x) = roles(x) \cup \{r\}$ и для $x' \in S \setminus \{x\}$ выполняется равенство $roles'(x') = roles(x')$
grant_right$(x, r, (y, \alpha_r))$	$x \in S$, $y \in E$, $(y, \alpha_r) \in P$ и $((y, own_r), r) \in de\_facto\_actions(x)$	$S' = S$, $E' = E$, $user' = user$, $roles' = roles$, $A' = A$, $H_E' = H_E$, $PA'(r) = PA(r) \cup \{(y, \alpha_r)\}$, и для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, s, write_t)$: $s \in (N_S \cup NF_S) \cap S$, $x \neq s$ и $r \in de\_facto\_roles(s)\}$, если $x \in LF_S \cap S$, то $F' = F$
create_first_session (u, r, y, z)	$u \in N_U$, $y \in E$, $z \notin E$, $(y, execute_r) \in PA(UA(u))$ и $r \in can\_manage\_rights(AUA(u))$	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $user'(z) = u$, для $s \in S$ выполняется равенство $user'(s) = user(s)$, $F' = F$, $roles'(z) = \varnothing$, для $s \in S$ выполняется равенство $roles'(s) = roles(s)$, $[z] = fa(u, y)$, $H_E'(z) = \varnothing$, для $e \in E$ выполняется равенство $H_E'(e) = H_E(e)$, $PA'(r) = PA(r) \cup \{(z, own_r)\}$, для $r' \in R \setminus \{r\}$ выполняется равенство $PA'(r') = PA(r')$
control(x, y, z)	$x, y \in S$, $x \neq y$, $z \in E$, $z \in [y]$ и или $x = z$, или $(x, z, write_m) \in F$	$S' = S$, $E' = E$, $PA' = PA$, $user' = user$, $roles' = roles$, $H_E' = H_E$, $A' = A \cup \{(x, y, own_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leq e\}$, если $x \in LF_S \cap S$, то $F' = F$

Продолжение таблицы

1	2	3
access_own(*x*, *y*)	$x, y \in S, x \neq y,$ $(y, own_r) \in de\_facto\_rights(x)$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E,$ $A' = A \cup \{(x, y, own_a)\},$ если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t): e \in E, x \neq e$ и $y \leq e\}$, если $x \in LF_S \cap S$, то $F' = F$
take_access_own(*x*, *y*, *z*)	$x, y, z \in S, x \neq z,$ $\{(x, y, own_a), (y, z, own_a)\} \subset A$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E,$ $A' = A \cup \{(x, z, own_a)\},$ если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, e, write_t): e \in E,$ $x \neq e$ и $z \leq e\}$, если $x \in LF_S \cap S$, то $F' = F$
access_write(*x*, *y*)	$x \in S, (y, write_r) \in$ $de\_facto\_rights(x)$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E,$ $A' = A \cup \{(x, y, write_a)\}$, если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup$ $\{(x, y, write_m)\} \cup \{(x, e, write_t): e \in E, x \neq e$ и $y \leq e\}$, если $x \in LF_S \cap S$, то $F' = F \cup \{(x, y, write_m)\}$
access_append(*x*, *y*)	$x \in S, (y, append_r) \in$ $de\_facto\_rights(x)$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, H_E' = H_E,$ $A' = A \cup \{(x, y, append_a)\},$ если $x \in (N_S \cup NF_S) \cap S$, то $F' = F \cup \{(x, y, write_m)\} \cup \{(x, e,$ $write_t): e \in E, x \neq e$ и $y \leq e\},$ если $x \in LF_S \cap S$, то $F' = F \cup \{(x, y, write_m)\}$
post(*x*, *y*, *z*)	$x, z \in S, y \in E, x \neq z,$ $(y, read_r) \in de\_facto\_rights(z)$ и или $(y, \alpha) \in de\_facto\_rights(x),$ где $\alpha \in \{write_r, append_r\},$ или $(x, y, \alpha) \in F,$ где $\alpha \in \{write_m, write_t\}$	$S' = S, E' = E, PA' = PA, user' = user, roles' = roles, A' = A, H_E' = H_E,$ если $\alpha \neq write_t$, то $F' = F \cup \{(x, z, write_m)\},$ если $\alpha = write_t$ и $x, z \in (N_S \cup NF_S) \cap S,$ то $F' = F \cup \{(x, z,$ $write_t)\},$ если $\alpha = write_t$ и $\{x, z\} \cap (LF_S \cap S) \neq \varnothing$, то $F' = F$

Зависимость условий и результатов применения правил преобразования состояний БР ДП-модели показана на рис. 1.

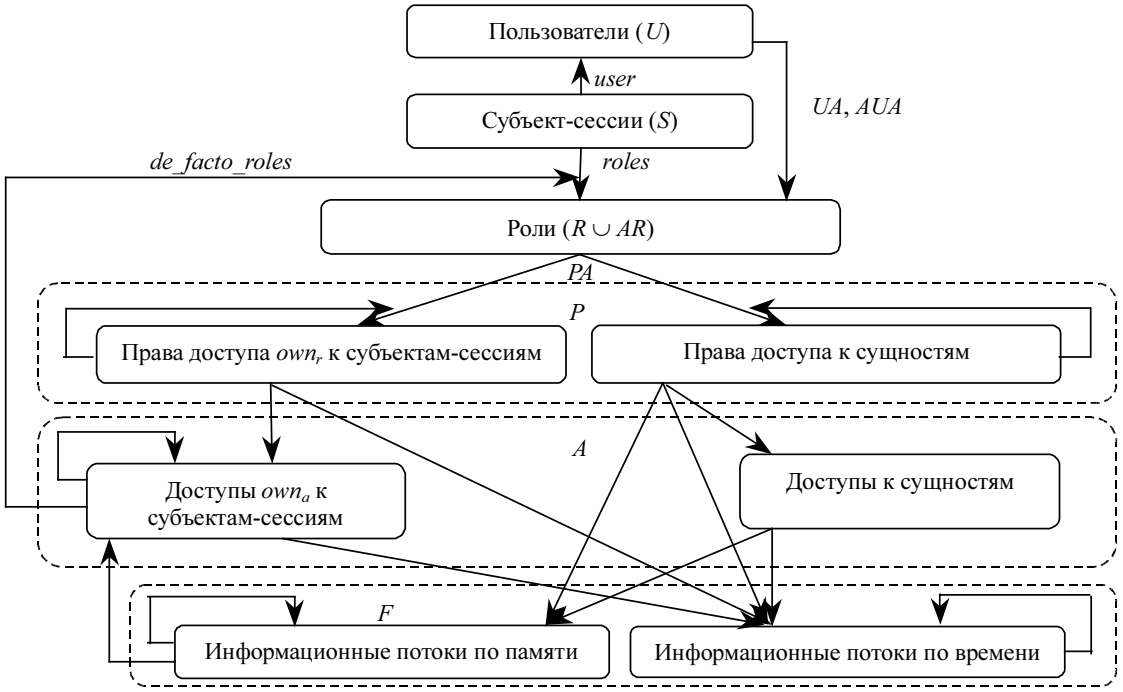


Рис. 1. Зависимость условий и результатов применения правил преобразования состояний

Проанализируем случай, когда для передачи права доступа ролей непосредственно взаимодействуют только две субъект-сессии.

Определение 12. Назовем траекторию функционирования системы $\Sigma(G^*, OP)$ траекторией без кооперации доверенных и недоверенных субъект-сессий для передачи прав доступа, если при ее реализации используются монотонные правила преобразования состояний, и доверенные субъект-сессии: не берут роли во множество текущих ролей, не дают другим ролям права доступа к сущностям, не получают доступ владения к субъект-сессиям.

Определение 13. Пусть $G_0 = (PA_0, user_0, roles_0, A_0, F_0, H_{E0})$ – состояние системы $\Sigma(G^*, OP)$, в котором существуют пользователь $u \in U$ и право доступа к сущности $(e, \alpha) \in P_0$. Определим предикат $can\_share((e, \alpha), u, G_0)$, который будет истинным тогда и только тогда, когда существуют состояния $G_1, \ldots, G_N = (PA_N, user_N, roles_N, A_N, F_N, H_{EN})$ и правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G_0 \tilde{A}_{op1} G_1 \tilde{A}_{op2} \ldots \tilde{A}_{opN} G_N$ является траекторией без кооперации доверенных и недоверенных субъект-сессий для передачи прав доступа, и существует субъект-сессия $x \in S_N$, такая, что $user_N(x) = u$, и право доступа к сущности $(e, \alpha) \in de\_facto\_rights_N(x)$, где $N \geq 0$.

Для упрощения записи алгоритмически проверяемых необходимых и достаточных условий истинности предиката $can\_share((y, \alpha), u, G_0)$ используем следующие определения.

Определение 14. Пусть $G = (PA, user, roles, A, F, H_E)$ – состояние системы $\Sigma(G^*, OP)$, в котором существуют недоверенный пользователь или недоверенная субъект-сессия $x \in N_U \cup (N_S \cap S)$, субъект-сессия или недоверенный пользователь $y \in N_U \cup S$. Определим предикат $directly\_access\_own(x, y, G)$, который будет истинным тогда и только тогда, когда выполняется одно из следующих условий 1 – 4.

Условие 1. Если $y \in N_U$ и $x \in N_U$, то существуют сущность $e_y \in E$ и роль $r_y \in R$, такие, что $(e_y, execute_r) \in PA(UA(y))$, $r_y \in can\_manage\_rights(AUA(y))$, и выполняется одно из условий: $(r_y \in UA(x))$, или $(x \in fa(y, e_y))$, или (существует сущность $e \in E$, такая, что $(e, \beta) \in PA(UA(x))$, где $\beta \in \{write_r, append_r, own_r\}$, и или $e \in fa(y, e_y)$, или $(e, \gamma) \in PA(UA(y))$, где $\gamma \in \{read_r, own_r\}$).

Условие 2. Если $y \in N_U$ и $x \in N_S \cap S$, то существуют сущность $e_y \in E$ и роль $r_y \in R$, такие, что $(e_y, execute_r) \in PA(UA(y))$, $r_y \in can\_manage\_rights(AUA(y))$, и выполняется одно из условий: или $(r_y \in UA(user(x)))$, или $(x \in fa(y, e_y))$, или (существует сущность $e \in E$, такая, что $(e, \beta) \in PA(UA(user(x))$, где $\beta \in \{write_r, append_r, own_r\}$, или $(x, e, write_m) \in F$, и или $e \in fa(y, e_y)$, или $(e, \gamma) \in PA(UA(y))$, где $\gamma \in \{read_r, own_r\}$).

Условие 3. Если $y \in S$ и $x \in N_U$, то выполняется одно из условий: или $((y, own_r) \in PA(UA(x)))$, или $(x \in [y])$, или (существует сущность $e \in E$, такая, что $(e, \beta) \in PA(UA(x))$, где $\beta \in \{write_r, append_r, own_r\}$, и или $e \in [y]$, или $y \in N_S \cap S$, $(e, \gamma) \in PA(UA(user(y)))$, где $\gamma \in \{read_r, own_r\}$, или $y \in L_S \cap S$, $(e, read_r) \in PA(roles(y))$, $(y, e) \notin y(E)$).

Условие 4. Если $y \in S$ и $x \in N_S \cap S$, то выполняется одно из условий: или $((y, own_r) \in PA(UA(user(x))))$, или $(x \in [y])$, или $((x, y, own_a) \in A)$, или (существует сущность $e \in E$, такая, что $(e, \beta) \in PA(UA(user(x))$, где $\beta \in \{write_r, append_r, own_r\}$, или $(x, e, write_m) \in F$, и или $e \in [y]$, или $y \in N_S \cap S$, $(e, \gamma) \in PA(UA(user(y)))$, где $\gamma \in \{read_r, own_r\}$, или $y \in L_S \cap S$, $(e, read_r) \in PA(roles(y))$, $(y, e) \notin y(E)$).

Определение 15. Пусть $G = (PA, user, roles, A, F, H_E)$ – состояние системы $\Sigma(G^*, OP)$, в котором недоверенная субъект-сессия или недоверенный пользователь $x \in N_U \cup (N_S \cap S)$, субъект-сессия или недоверенный пользователь $y \in N_U \cup S$. Определим предикат $directly\_grant\_right(x, y, G)$, который будет истинным тогда и только тогда, когда выполняется одно из следующих условий 1 – 4.

Условие 1. Если $y \in N_U$ и $x \in N_U$, то существует роль $r \in can\_manage\_rights(AUA(x)) \cap UA(y)$.

Условие 2. Если $y \in N_U$ и $x \in N_S \cap S$, то существуют роль $r \in can\_manage\_rights(AUA(user(x))) \cap UA(y)$.

Условие 3. Если $y \in S$ и $x \in N_U$, то выполняется одно из условий:

- $y \in N_S \cap S$ и существуют роль $r \in can\_manage\_rights(AUA(x)) \cap UA(user(y))$;
- $y \in L_S \cap S$ и существуют роль $r \in can\_manage\_rights(AUA(x)) \cap roles(y)$.

Условие 4. Если $y \in S$ и $x \in N_S \cap S$, то выполняется одно из условий:

- $y \in N_S \cap S$ и существуют роль $r \in can\_manage\_rights(AUA(user(x))) \cap UA(user(y))$.
- $y \in L_S \cap S$ и существуют роль $r \in can\_manage\_rights(AUA(user(x))) \cap roles(y)$.

Возможно доказать утверждения 2 и 3, в которых обосновываются достаточные условия истинности предиката $can\_share((e, \alpha), x, G_0)$ для случая, когда при передаче прав доступа ролей взаимодействуют только две субъект-сессии двух пользователей.

Утверждение 2. Пусть $G_0 = (PA_0, user_0, roles_0, A_0, F_0, H_{E0})$ – состояние системы $\Sigma(G^*, OP)$, в котором существуют недоверенный пользователь или недоверенная субъект-сессия $x \in N_U \cup (N_S \cap S_0)$, субъект-сессия или недоверенный пользователь $y \in N_U \cup S_0$ и право доступа к сущности $(e, \alpha) \in P_0$. Пусть истинен предикат $directly\_access\_own(x, y, G_0)$ и выполняется одно из условий:

- если $y \in N_U$, то $(e, \alpha) \in PA_0(UA_0(y))$;
- если $y \in N_S \cap S_0$, то $(e, \alpha) \in PA_0(UA_0(user_0(y)))$;
- если $y \in L_S \cap S_0$, то $(e, \alpha) \in PA_0(roles_0(y))$.

Тогда выполняется одно из следующих условий.

Условие 1. Если $x \in N_U$, то истинен предикат $can\_share((e, \alpha), x, G_0)$.

Условие 2. Если $x \in N_S \cap S_0$, то истинен предикат $can\_share((e, \alpha), user_0(x), G_0)$.

Утверждение 3. Пусть $G_0 = (PA_0, user_0, roles_0, A_0, F_0, H_{E0})$ – состояние системы $\Sigma(G^*, OP)$, в котором существуют недоверенный пользователь или недоверенная субъект-сессия $x \in N_U \cup (N_S \cap S_0)$, субъект-сессия

или недоверенный пользователь $y \in N_U \cup S_0$ и право доступа к сущности $(e, \alpha) \in P_0$. Пусть истинен предикат *directly_grant_right*(x, y, G_0) и выполняется одно из условий:

- если $x \in N_U$, то $(e, own_r) \in PA_0(UA_0(x))$;
- если $x \in N_S \cap S_0$, то $(e, own_r) \in PA_0(UA_0(user_0(x)))$.

Тогда выполняется одно из условий.

Условие 1. Если $y \in N_U$, то истинен предикат *can_share*$((e, \alpha), y, G_0)$.

Условие 2. Если $y \in S_0$, то истинен предикат *can_share*$((e, \alpha), user_0(y), G_0)$.

Таким образом, с применением БР ДП-модели возможен анализ условий передачи прав доступа ролей в КС с РУД. В дальнейшем возможно обоснование условий возникновения в КС с РУД информационных потоков по памяти или по времени.

ЛИТЕРАТУРА

1. *Девянин П.Н.* Модели безопасности компьютерных систем: Учеб. пособие для студ. высш. учеб. заведений. М.: Издательский центр «Академия», 2005. 144 с.
2. *Bishop M.* Computer Security: Art and Science. 2002. 1084 p.
3. *Sandhu R.* Role-Based Access Control, Advanced in Computers // Academic Press. 1998. V. 46.
4. *Девянин П.Н.* Анализ безопасности управления доступом и информационными потоками в компьютерных системах. М.: Радио и связь, 2006. 176 с.

2008 Математические основы компьютерной безопасности № 1(1)
DOI 10.17223/20710410/1/12
УДК 004.94

ПРИМЕНЕНИЕ ДП-МОДЕЛЕЙ ДЛЯ АНАЛИЗА ЗАЩИЩЕННОСТИ СЕТЕЙ

Д.Н. Колегов

Томский государственный университет

E-mail: kden@sibmail.com

В статье вводится понятие замыкания ДП-моделей. Описываются и обосновываются алгоритмы их построения. Рассматривается применение таких моделей для анализа защищенности компьютерных сетей. Предлагаются методы анализа путей распространения прав доступа и информационных потоков. В рамках теории ДП-моделей формализуются модели топологического анализа защищенности REM и VTG.

Ключевые слова: *ДП-модель*, *ФАС ДП-модель*, *модель Take-Grant*, *замыкание*, *топологический анализ защищенности*, *графы атак*, *моделирование атак.*

В современных методологиях анализа защищенности компьютерных систем (КС) учитывается «топология» КС – взаимосвязь объектов и субъектов КС, их свойств и характеристик. Такой анализ защищенности называется *топологическим*, а автоматизированные средства, его выполняющие, *топологическими сканерами безопасности*.

Основной математической моделью топологического анализа защищенности является граф атак, который строится на основе модели нарушителя, конфигурации сети (правил фильтрации межсетевого экрана (МЭ), маршрутизации, обнаружения и предотвращения атак, достижимости хостов и т.д.), результатов сканирования сети, анализа уязвимостей или теста на вторжение. Центральными задачами являются синтез графа атак и его анализ (вероятностный, минимизационный и т.д.).

Граф атак содержит все известные траектории (сценарии, пути) атак реализации нарушителем угроз. Результатом его анализа может являться:

- перечень успешных атак, не обнаруживаемых IDS;
- соотношение реализуемых мер безопасности и уровня защищенности сети;
- перечень наиболее критичных уязвимостей;
- перечень мер, позволяющих предотвратить использование уязвимостей в ПО, для которого отсутствуют обновления;
- наименьшее множество мер, реализация которых сделает сеть защищенной.

Графы атак также используются при расследовании компьютерных инцидентов, для анализа рисков и корреляций предупреждений систем обнаружения атак. Ключевой проблемой построения графа атак является масштабируемость – возможность построения графа атак для сети с большим числом хостов и уязвимостей.

В настоящее время представлено значительное многообразие подходов к синтезу и анализу графов атак [1]. Большинство работ посвящено синтезу эффективных методов построения графов атак и лишь единицы – формальным моделям анализа современных КС, исследованию и математическому обоснованию их свойств. Так, Shahriary, Sadoddin, Jalili и Zakeri [2] использовали положения классической модели Take-Grant для синтеза полиномиального алгоритма построения графа атак путем построения замыкания графа доступов модели Take-Grant. В то же время известно [3], что модель Take-Grant является неадекватной для анализа безопасности современных КС. Одной из современных моделей анализа безопасности КС с дискреционным управлением доступа является развитие модели Take-Grant – ДП-модель и ее расширения.

Основой всех ДП-моделей является базовая ДП-модель. В ней моделируемая КС представляется абстрактной иерархической системой, состоящей из сущностей, каждое состояние которой представляется графом доступов. Переход из состояния в состояние осуществляется в результате применения одного из правил преобразования графа доступа: *take_right*(), *grant_right*(), *remove_right*(), *own_take*(), *own_remove*(), *create_subject*(), *delete_entity*(), *rename_entity*(), *access_read*(), *access_write*(), *access_append*(), *flow*(), *find*(), *post*(), *pass*() и *control*(). Для описания условий возникновения информационных потоков $write_m$, $write_t$ между сущностями используются права доступа $read_r$, $write_r$, $append_r$, $execute_r$, own_r и виды доступа $read_a$, $write_a$, $append_a$. Для описания условий передачи между сущностями КС прав доступа используется право доступа own_r. Субъекты могут создавать новые сущности и получать к ним любые права доступа.

С помощью ДП-модели анализируются условия передачи прав доступа и реализации информационных потоков по памяти и по времени при кооперации субъектов и без кооперации доверенных и недоверенных субъектов, а также предлагаются методы их предотвращения. На основе базовой ДП-модели путем изменения правил преобразования состояний строятся: ДП-модель без кооперации субъектов (БК ДП-модель), ДП-

модель с блокирующими доступами доверенных субъектов (БД ДП-модель) и ДП-модель с функционально-ассоциированными сущностями (ФАС ДП-модель). Последняя позволяет анализировать условия получения прав доступа к объекту при реализации информационных потоков по памяти, что и позволяет успешно использовать ее при анализе защищенности современных КС.

1. Обозначения и определения

В соответствии с [3] будем использовать следующие обозначения и определения: $E = O \cup C$ – множество сущностей, где O – множество объектов, C – множество контейнеров и $O \cap C = \varnothing$; $S \subseteq E$ – множество субъектов; $R_r = \{read_r, write_r, append_r, execute_r, own_r\}$ – множество видов прав доступа, где $read_r$ – право доступа на чтение из сущности, $write_r$ – право доступа на запись в сущность, $append_r$ – право доступа на запись в конец сущности, $execute_r$ – право доступа на выполнение (активизацию) сущности, own_r – право доступа на владение сущностью, позволяющее субъекту-владельцу передавать права доступа к сущности другим субъектам или удалить сущность; $R_a = \{read_a, write_a, append_a\}$ – множество видов доступа, где $read_a$ – доступ на чтение из сущности, $write_a$ – доступ на запись в сущность, $append_a$ – доступ на запись в конец слова, содержащегося в сущности; $R_f = \{write_m, write_t\}$ – множество видов информационных потоков, где $write_m$ – информационный поток по памяти на запись в сущность, $write_t$ – информационный поток по времени на запись в сущность; $R_{raf} = R_r \cup R_a \cup R_f$ – множество видов прав доступа, видов доступа и видов информационных потоков, при этом множества R_r, R_a, R_f попарно не пересекаются.

Для заданного на E отношения частичного порядка $\leq$ функция H: $E \to 2^E$ называется *функцией иерархии сущностей*, если она удовлетворяет условиям:

1) для любой сущности $e \in H(c)$ имеет место $e < c$ и не существует сущности-контейнера $d \in C$, такой, что $e < d$ и $d < c$;

2) для любых сущностей $e_1, e_2 \in E$ если $e_1 \neq e_2$, то $H(e_1) \cap H(e_2) = \varnothing$, и если $e_1 < e_2$, то или $e_1, e_2 \in E \setminus S$, или $e_1, e_2 \in S$;

3) $H(o) = \varnothing$ для любого $o \in O$; $H(e) \subset E \setminus S$ для любого $e \in E \setminus S$; $H(s) \subset S$ для всех $s \in S$.

Пусть определены множества S, E, $R \subseteq S \times E \times R_r$, $A \subseteq S \times E \times R_a$, $F \subseteq E \times E \times R_f$ и функция иерархии сущностей H. Конечный помеченный ориентированный граф без петель $G = (S, E, R \cup A \cup F, H)$, где элементы множеств S, E являются вершинами графа, а элементы множества $R \cup A \cup F$ ребрами, называют *графом доступов*.

Пусть также $\Sigma(G^*, OP)$ есть система, каждое состояние которой представляется графом доступов и где G^* – множество всех возможных состояний, OP – множество правил преобразования состояний [3]. Обозначим $G \vdash_{op} G'$ – переход системы $\Sigma(G^*, OP)$ из состояния G в состояние G' с использованием правила преобразования состояний $op \in OP$ и $\Sigma(G^*, OP, G_0)$ – систему $\Sigma(G^*, OP)$ с начальным состоянием G_0.

Пусть, наконец, подмножества $L_a \subset S \times O \times R_a$, $L_r \subset S \times O \times R_r$, $L_f \subset O \times O \times R_f$ являются множествами соответственно разрешенных доступов субъектов к объектам, разрешенных прав доступа субъектов к объектам, разрешенных информационных потоков между объектами и подмножества $N_a \subset S \times O \times R_a$, $N_r \subset S \times O \times R_r$ и $N_f \subset O \times O \times R_f$ суть множества соответственно запрещенных доступов субъектов к объектам, запрещенных прав доступа субъектов к объектам и запрещенных информационных потоков между объектами КС. При этом выполняются условия: $L_a \cap N_a = \varnothing$, $L_a \cup N_a = S \times O \times R_a$, $L_r \cap N_r = \varnothing$, $L_r \cup N_r = S \times O \times R_r$, $L_f \cap N_f = \varnothing$ и $L_f \cup N_f = O \times O \times R_f$.

Сущность называется *функционально-ассоциированной* с субъектом, если от нее зависит вид преобразования данных, реализуемого субъектом в данном или некотором последующем состоянии системы $\Sigma(G^*, OP)$.

Элементы множеств N_a, N_r, N_f будем называть *угрозами* нарушений безопасности в КС. *Нарушением безопасности* в КС называют ее переход в состояние, в котором либо получен запрещенный доступ из N_a, либо произошла утечка запрещенного права доступа из N_r, либо реализован запрещенный информационный поток из N_f.

2. Построение замыкания ДП-моделей

Для проверки истинности предикатов *can_share*(), *can_write_memory*(), *can_steal*() и др. предикатов в ДП-моделях [3] наиболее эффективно использовать алгоритмы, позволяющие осуществлять проверку истинности указанных предикатов для всех пар вершин одновременно и проводить анализ путей распространения прав доступа и информационных потоков. Такие алгоритмы реализуют преобразование графа доступов в его замыкание.

При построении замыкания графа доступов необходимо выполнить два стандартных для данной задачи шага [4]. Первый шаг направлен на устранение возможности применения немонотонных правил преобразования состояний и обосновывается тем, что при анализе условий передачи прав доступа, реализации информационных потоков по памяти и по времени возможно использование только монотонных правил преобразования состояний [3]. Поэтому при построении замыкания графа доступов немонотонные правила можно

исключить из рассмотрения. После этого граф доступов может только расти. Для того чтобы число вершин в графе не увеличивалось до бесконечности, каждый субъект должен иметь объект с правом владения на него. Этого будет достаточно для передачи субъектом прав доступа, создания субъектов или реализации информационных потоков, после чего в графе число вершин увеличиваться не будет, а число ребер между парой вершин будет ограничено числом прав доступа, видов доступа и информационных потоков.

На основе схемы построения замыкания графа доступов модели Take-Grant [5] дадим определения, построим и обоснуем алгоритмы замыкания графа доступов ДП-моделей.

Пусть $G = (S, E, R \cup A \cup F, H)$ – граф доступов, где для всех $s \in S$, $z \in E \setminus S$, таких, что $(s, z, write_r) \in R$ или $(s, z, append_r) \in R$, существует $y \in E$, что $y < z$ и $(s, y, own_r) \in R$. Тогда *own-замыканием* графа G будем называть граф $G^{own} = (S, E, R^{own} \cup A \cup F^{own}, H)$, полученный из G применением последовательности правил *take_right*() и *grant_right*(), если при этом выполнены следующие условия:

- для всех ребер $(a, b, \alpha) \in R^{own} \setminus R$: $\alpha = own_r$;
- для всех ребер $(c, d, \beta) \in F^{own} \setminus F$: $\beta = write_t$;
- все ребра $(c, d, \beta) \in F^{own} \setminus F$ получены в результате применения правил *take_right*() и *grant_right*();
- применение последовательности правил *take_right*() и *grant_right*() не приводит к появлению новых ребер указанных видов.

Пусть $G = (S, E, R \cup A \cup F, H)$ – граф доступов, где для всех $s \in S$, $z \in E \setminus S$, таких, что $(s, z, write_r) \in R$ или $(s, z, append_r) \in R$, существует $y \in E$, что $y < z$ и $(s, y, own_r) \in R$. Тогда *tgo-замыканием* графа G будем называть граф $G^{tgo} = (S, E, R^{tgo} \cup A \cup F^{tgo}, H)$, полученный из G применением последовательности правил *take_right*(), *grant_right*() и *own_take*(), если при этом выполнены условия:

- для всех ребер $(a, b, \alpha) \in R^{tgo} \setminus R$: $\alpha \in R_r$;
- для всех ребер $(c, d, \beta) \in F^{tgo} \setminus F$: $\beta = write_t$;
- все ребра $(c, d, \beta) \in F^{tgo} \setminus F$ получены в результате применения правил *take_right*(), *grant_right*() и *own_take*();
- применение последовательности правил *take_right*(), *grant_right*() и *own_take*() не приводит к появлению новых ребер указанных видов.

Пусть $G = (S, E, R \cup A \cup F, H)$ – граф доступов, где для всех $s \in S$, $z \in E \setminus S$, таких, что $(s, z, write_r) \in R$ или $(s, z, append_r) \in R$, существует $y \in E$, что $y < z$ и $(s, y, own_r) \in R$. Тогда *замыканием* (*access-замыканием*) графа G будем называть граф $G^{access} = (S, E, R^{access} \cup A^{access} \cup F^{access}, H)$, полученный из G применением последовательности правил *take_right*(), *grant_right*(), *own_take*(), *access_read*(), *access_write*(), *access_append*(), *flow*(), *find*(), *post*() и *pass*(), если при этом применение к графу G^{access} указанных правил не приводит к появлению в нем новых ребер.

Пусть $G = (S, E, R \cup A \cup F, H)$ – граф доступов, где для всех $s \in S$, $z \in E \setminus S$, таких, что $(s, z, write_r) \in R$ или $s, z, append_r) \in R$, существует $y \in E$, что $y < z$ и $(s, y, own_r) \in R$. Тогда *control-замыканием* графа G будем называть граф $G^{control} = (S, E, R^{control} \cup A^{control} \cup F^{control}, H)$, полученный из G применением последовательности правил *take_right*(), *grant_right*(), *own_take*(), *access_read*(), *access_write*(), *access_append*(), *flow*(), *find*(), *post*(), *pass*() и *control*(), если при этом применение к графу $G^{control}$ указанных правил не приводит к появлению в нем новых ребер.

Опишем и обоснуем алгоритмы построения замыкания графов доступов ДП-моделей.

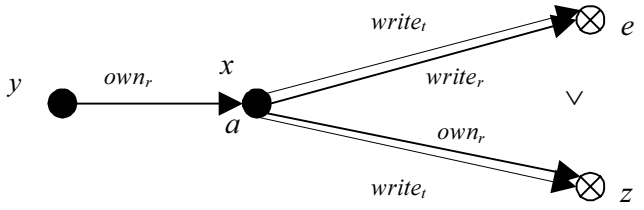
2.1. Построение замыкания графа доступов базовой ДП-модели

Алгоритм 1 построения *own*-замыкания графа доступов.

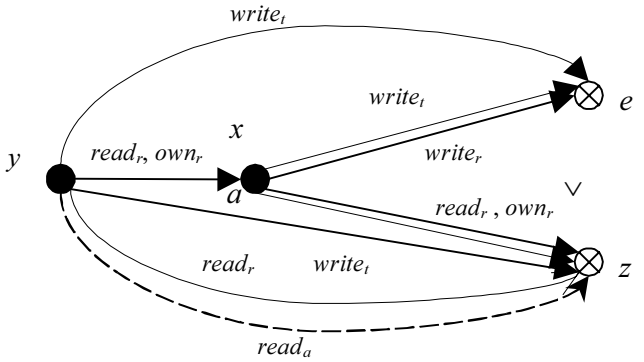
1. Для всех $s \in S$, $z \in E \setminus S$, таких, что $(s, z, write_r) \in R$ или $(s, z, append_r) \in R$, последовательно применить правила *create_entity*(x, y, z), *own_take*$(execute_r, x, y)$, *create_subject*(x, v, y). Новые сущности и ребра занести в S, E, R и F, определить H с учетом изменения иерархии сущностей.
2. $L = \{(x, y, own_r) \in R\}$, $N = \varnothing$.
3. Выбрать из списка L первое ребро (x, y, own_r), занести x и y в N, удалить ребро (x, y, own_r) из L.
4. Для всех вершин $z \in N$ проверить возможность применения правил *take_right*() и *grant_right*() на вершинах x, y, z с использованием ребра (x, y, own_r). Если в результате применения правил получаются новые ребра вида (a, b, own_r), где $\{a, b\} \subset \{x, y, z\}$, то занести их в конец L и R^{own}, а ребра вида $(a, b, write_t)$ в F^{own}.
5. Если L не пусто, перейти на шаг 2.

Замечание. При построении замыкания графа доступов в модели Take-Grant первый шаг алгоритма построения *tg*-замыкания заключается в применении правила *create*$(s, o, \{t, g, r, w\})$ для каждого $s \in S$, что достаточно для анализа передачи прав доступа и информационных потоков. Так как ДП-модель является иерархической, то для создания сущности z в сущности-контейнере e субъектом x необходимо, чтобы $(x, e, write_r) \in R$ или $(x, e, append_r) \in R$. При этом будут реализованы соответствующие информационные потоки по времени. В состоянии, изображенном на рис.1, субъект y может сначала получить права доступа к сущно-

сти-контейнеру e, а затем с помощью них создать сущность. Однако можно видеть, что субъект y может получить все права доступа к сущности z и реализовать все возможные информационные потоки по времени, поэтому для субъекта y нет необходимости осуществлять правило преобразования *create_entity*(). Точно так же в случае с созданием субъектов на шаге 1 *create_subject*(x, v, y) можно реализовать информационные потоки по времени от v к y с помощью правила *post*(y, x, v). Таким образом, шаг 1 в алгоритме является достаточным.



+ *own_take* ($read_r$, x, z), *take_right* ($read_r$, y, x, z), *access_read*(y, z)



+ *post* (y, x, v)

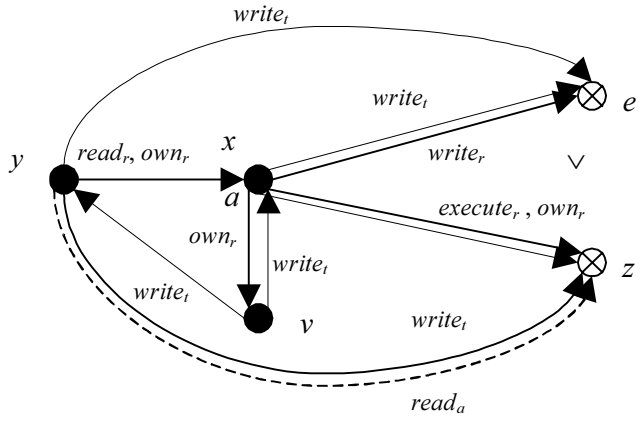


Рис. 1

Теорема 1. Алгоритм 1 корректно строит *own*-замыкание графа доступов.

Доказательство. Пусть $G' = (S, E, R' \cup A \cup F', H)$ – граф доступов, полученный из $G = (S, E, R \cup A \cup F, H)$ в результате применения алгоритма 1. Пусть $G^{own} = (S, E, R^{own} \cup A \cup F^{own}, H)$ есть *own*-замыкание графа G. Необходимо доказать, что $G' = G^{own}$. Покажем, что $R^{own} = R'$. Доказательство проведем методом от противного. Пусть существует ребро $(x, y, own_r) \in R^{own} \setminus R'$. Тогда существуют ребра (a, b, own_r), (c, d, own_r) из R^{own}, применение к которым правила *take_right* или *grant_right* привело к появлению ребра (x, y, own_r). Если (a, b, own_r), $(c, d, own_r) \in R'$, то, согласно описанию алгоритма 1, $(x, y, own_r) \in R'$ и получаем противоречие. Значит, или $(a, b, own_r) \in R^{own} \setminus R'$, или $(c, d, own_r) \in R^{own} \setminus R'$. Вместе с этим граф G^{own} получен из G в результате применения конечной последовательности правил *take_right*() и *grant_right*(), и ребра (a, b, own_r), (c, d, own_r) должны быть получены раньше, чем ребро (x, y, own_r). Проводя рассуждения подобным образом для

ребер, с использованием которых получены ребра (a, b, own_r), (c, d, own_r), придем к противоречию в связи с тем, что все ребра графа G изначально содержатся в графе G'. Таким образом, $R^{own} = R'$. А так как ребро $(x, y, write_t)$ получается в результате применений правил *take_right*() или *grant_right*(), то и $F^{own} = F'$. ■

Алгоритм 2 построения *tgo*-замыкания графа доступов.

1. Выполнить алгоритм 1.
2. Для всех ребер вида $(x, y, own_r) \in R$ применить правило *own_take*(α_r, x, y), $\alpha_r \in R_r \setminus \{own_r\}$, и если полученное ребро $(x, y, \alpha_r) \notin R$, занести его в R^{tgo}.
3. Для каждой пары ребер вида (x, y, own_r), $(y, z, \alpha_r) \in R$ применить правило *take_right*(α_r, x, y, z), и если полученное ребро $(x, z, \alpha_r) \notin R$, то занести его в R^{tgo}, а ребро $(y, x, write_t)$ – в F^{tgo}.
4. Для каждой пары ребер вида (x, y, own_r), $(x, z, \alpha_r) \in R$ применить правило *grant_right*(α_r, x, y, z), и если полученное ребро $(y, z, \alpha_r) \notin R$, то занести его в R^{tgo}, а ребро $(x, y, write_t)$ – в F^{tgo}.
5. Для каждой пары ребер вида (x, y, own_r), $(y, z, \alpha_r) \in R$ применить правило *take_right*(α_r, x, y, z), и если полученное ребро $(x, z, \alpha_r) \notin R$, то занести его в R^{tgo}, а ребро $(y, x, write_t)$ – в F^{tgo}.

Теорема 2. Алгоритм 2 корректно строит *tgo*-замыкание графа доступов.

Доказательство. Для построения *tgo*-замыкания графа доступов всем субъектам графа доступов после выполнения алгоритма 1 необходимо выполнить следующие шаги:

1) используя правило *own_take*(), забрать права доступа;

2) используя правило *grant_right*(), раздать права доступа и, используя правило *take_right*(), забрать права доступа;

3) используя правило *take_righ*()*t*, раздать права доступа и, используя правило *grant_right*(), забрать права доступа;

Объединяя шаги 2 – 3, получаем алгоритм 2. ■

Алгоритм 3 построения *access*-замыкания графа доступов.

1. Выполнить алгоритм 2.
2. Для всех ребер вида $(x, z, write_r) \in R$, где $x \in S$, а $y, z \in E$ и $y \in H(z)$, применить правило *rename_entity*(x, y, z), все полученные новые ребра занести в F^a.
3. Для всех ребер вида $(x, y, read_r) \in R$, $(x, y, read_a) \notin A$ применить правило *access_read*(x, y), полученные новые ребра $(x, y, read_a)$ и $(y, x, write_m)$, $\{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leq e\}$ занести в A^a и F^a соответственно.
4. Для всех ребер вида $(x, y, write_r) \in R$, $(x, y, write_a) \notin A$ применить правило *access_write*(x, y), полученные ребра $(x, y, write_a)$ и $(x, y, write_m)$, $\{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leq e\}$ занести в A^a и F^a соответственно.
5. Для всех ребер вида $(x, y, append_r) \in R$, $(x, y, append_a) \notin A$ применить правило *access_append*(x, y), полученные ребра $(x, y, append_a)$ и $(x, y, write_m)$, $\{(x, e, write_t)$: $e \in E$, $x \neq e$ и $y \leq e\}$ занести в A^a и F^a соответственно.
6. $L = \{(x, y, \alpha) \in R \cup F\}$, $N = \varnothing$.
7. Выбрать из списка L первое ребро (x, y, α), занести x и y в N, удалить ребро (x, y, α) из L.
8. Для всех вершин $z, y' \in N$ проверить возможность применения правила *flow*() на четверке вершин x, y, y', z с использованием ребер (x, y, α) и (z, y', β) и правил *post*(), *pass*(), *find*() на тройке вершин x, y, z с использованием ребра (x, y, α). Если в результате применения правил получаются новые ребра вида (a, b, γ), где $\{a, b\} \subset \{x, y, y', z\}$, $\gamma \in \{write_m, write_t\}$, то занести их в конец L и F^a.
9. Если L не пусто, перейти на шаг 6.

Теорема 3. Алгоритм 3 корректно строит *access*-замыкание графа доступов.

Доказательство. После выполнения шагов 1 – 5 алгоритма 3 в графе доступов будут все ребра вида $(x, y, \alpha) \in A$, где $\alpha \in \{read_a, write_a, append_a\}$. Шаги 5 – 8 алгоритма 3 аналогичны шагам алгоритма 1, в силу корректности которого по теореме 1 приходим к истинности теоремы 3. ■

2.2. Построение замыкания графа доступов БК ДП-модели

Перечислим основные положения БК ДП-модели:

- субъект $u \in L_S$ называют *доверенным*, если он обладает правом доступа владения к каждой сущности системы;

- каждый субъект системы $\Sigma(G^*, OP)$ является либо доверенным, либо недоверенным;

- доверенные субъекты системы $\Sigma(G^*, OP)$ не участвуют в реализации информационных потоков по времени, т.е. во всех состояниях отсутствуют информационные потоки по времени, исходящие из доверенных субъектов;

- правила преобразования состояний *take_right*(α_r, x, y, z), *grant_right*(α_r, x, y, z) могут быть использованы только недоверенными субъектами.

В соответствии с данными положениями модифицируются условия и результаты применения правил преобразования состояний базовой ДП-модели [3, табл. 2.1]. Опишем и обоснуем алгоритмы построения замыкания графа доступов БК ДП-модели.

Алгоритм 4 построения *own*-замыкания графа доступов.

1. Для всех $s \in S$, $z \in E \setminus S$, таких, что $(s, z, write_r) \in R$ или $(s, z, append_r) \in R$, последовательно применить правила *create_entity*(x, y, z), *own_take*$(execute_r, x, y)$, *create_subject*(x, v, y). Новые сущности и ребра занести в S, E, R и F, определить H с учетом изменения иерархии сущностей.
2. $L = \{(x, y, own_r) \in R\}$, где $x \in N_S$, $N = \varnothing$.
3. Выбрать из списка L первое ребро (x, y, own_r), занести x и y в N, удалить ребро (x, y, own_r) из L.
4. Для всех вершин $z \in N \cap N_S$ проверить возможность применения правил *take_right* и *grant_right* на вершинах x, y, z с использованием ребра (x, y, own_r). Если в результате применения правил получаются новые ребра вида (a, b, own_r), где $\{a, b\} \subset \{x, y, z\}$, то занести их в конец L и R^{own}, а ребра вида $(a, b, write_t)$ для $a \in N_S$ – в F^{own}.
5. Если L не пусто, перейти на шаг 2.

Теорема 4. Алгоритм 1 корректно строит *own*-замыкание графа доступов.

Доказательство теоремы выполняется аналогично доказательству теоремы 1 с учетом того, что только недоверенные субъекты могут инициировать выполнение правил преобразования состояний *take_right*()и *grant_right*(). ■

Алгоритм 5 построения *tgo*-замыкания графа доступов.

1. Выполнить алгоритм 4.
2. Для всех ребер вида $(x, y, own_r) \in R$ применить правило *own_take* (α_r, x, y), $\alpha_r \in R_r \setminus \{own_r\}$; если полученное ребро $(x, y, \alpha_r) \notin R$, занести его в R^{tgo}.
3. Для каждой пары ребер вида (x, y, own_r), $(y, z, \alpha_r) \in R$, где $x \in N_S$, применить правило *take_right* (α_r, x, y, z), и если полученное ребро $(x, z, \alpha_r) \notin R$, то занести его в R^{tgo}, а ребро $(y, x, write_t)$, где $y \in N_S$, – в F^{tgo}.
4. Для каждой пары ребер вида (x, y, own_r), $(x, z, \alpha_r) \in R$, где $x \in N_S$, применить правило *grant_right* (α_r, x, y, z), и если полученное ребро $(y, z, \alpha_r) \notin R$, то занести его в R^{tgo}, а ребро $(x, y, write_t)$, где $x \in N_S$, – в F^{tgo}.
5. Для каждой пары ребер вида (x, y, own_r), $(y, z, \alpha_r) \in R$, где $x \in N_S$, применить правило *take_right* (α_r, x, y, z), и если полученное ребро $(x, z, \alpha_r) \notin R$, то занести его в R^{tgo}, а ребро $(y, x, write_t)$, где $y \in N_S$, – в F^{tgo}.

Теорема 5. Алгоритм 5 корректно строит *tgo*-замыкание графа доступов.

Доказательство теоремы выполняется аналогично доказательству теоремы 2 с учетом того, что инициировать выполнение правила *own_take* могут все субъекты, а правил *take_right*() и *grant_right*() – только недоверенные субъекты. ■

Алгоритм 6 построения *access*-замыкания графа доступов.

1. Выполнить алгоритм 5.
2. Для всех ребер вида $(x, z, write_r) \in R$, где $x \in N_S$, а $y, z \in E$ и $y \in H(z)$, применить правило *rename_entity*(x, y, z), при получении новых ребер занести их в F^a.
3. Для всех ребер вида $(x, y, read_r) \in R$, $(x, y, read_a) \notin A$ применить правило *access_read*(x, y), полученные ребра $(x, y, read_a)$ и $(y, x, write_m)$ занести в A^a и F^a соответственно. Если $x \in N_S$, то $F^a = F^a \cup \{(x, e, write_t)$: $e \in E, x \neq e$ и $y \leq e\}$.
4. Для всех ребер вида $(x, y, write_r) \in R$, $(x, y, write_a) \notin A$ применить правило *access_write*(x, y), полученные ребра $(x, y, write_a)$ и $(x, y, write_m)$ занести в A^a и F^a соответственно. Если $x \in N_S$, то $F^a = F^a \cup \{(x, e, write_t)$: $e \in E, x \neq e$ и $y \leq e\}$.
5. Для всех ребер вида $(x, y, append_r) \in R$, $(x, y, append_a) \notin A$ применить правило *access_append*(x, y), полученные ребра $(x, y, append_a)$ и $(x, y, write_m)$ занести в A^a и F^a соответственно. Если $x \in N_S$, то $F^a = F^a \cup \{(x, e, write_t)$: $e \in E, x \neq e$ и $y \leq e\}$.
6. $L = \{(x, y, \alpha) \in R \cup F\}$, $N = \varnothing$.
7. Выбрать из списка L первое ребро (x, y, α), занести x и y в N, удалить ребро (x, y, α) из L.
8. Для всех вершин $z, y' \in N$ проверить возможность применения правила *flow* на четверке вершин x, y, y', z с использованием ребер (x, y, α) и (z, y', β) и правил *post*, *pass*, *find* на тройке вершин x, y, z с использованием ребра (x, y, α). Новые ребра вида (a, b, γ), где $\{a, b\} \subset \{x, y, y', z\}$, $\gamma \in \{write_m, write_t\}$, получающиеся в зависимости от правила преобразования состояния БК ДП-модели и принадлежности субъектов к множествам L_S и N_S, занести в конец L и F^a.
9. Если L не пусто, перейти на шаг 6.

Теорема 6. Алгоритм 6 корректно строит *access*-замыкание графа доступов.

Доказательство теоремы выполняется аналогично доказательству теоремы 3 с учетом правил преобразования БК ДП-модели. ■

2.3. Построение замыкания графа доступов БД ДП-модели

Перечислим основные положения и некоторые обозначения БД ДП-модели:

- любой доступ любого доверенного субъекта к сущности называется блокирующим;
- $E_B \subset E$ – множество сущностей, к которым имеют доступ доверенные субъекты системы $\Sigma(G^*, OP)$;
- H_B: $E \to 2^E$ – функция иерархии сущностей с учетом блокирующих доступов доверенных субъектов, сопоставляющая каждой сущности $c \in E$ множество сущностей $H_B(c) \subset H(c)$, удовлетворяющее условию $H_B(c) = \{e \in H(c): e <_B c\}$, где $<_B$ есть заданное на множестве сущностей E отношение частичного порядка;
- блокирующие доступы доверенных субъектов к сущностям системы $\Sigma(G^*, OP)$ препятствуют реализации информационных потоков по времени с использованием данных сущностей и иерархии сущностей, в которую входит данная сущность, за исключением случая, когда сущность является субъектом и участвует в реализации данного информационного потока по времени.

В соответствии с данными положениями модифицируются условия и результаты применения правил преобразования состояний базовой ДП-модели и БК ДП-модели [3, табл. 3.2].

Опишем и обоснуем алгоритмы построения замыкания графа доступов БД ДП-модели. Для построения *own*-замыкания и *tgo*-замыкания графа доступов БД ДП-модели применяются алгоритмы 4 и 5, так как в соответствии с положениями БД ДП-модели правила *take_right*(), *grant_right*(), *own_take*() применяются с условиями и результатами табл. 2.1 БК ДП-модели из [3].

Алгоритм 7 построения *access*-замыкания графа доступов.

1. Выполнить алгоритм 5.
2. Для всех ребер вида $(x, z, write_r) \in R$, где $x \in N_S$, а $y, z \in E$ и $y \in H(z)$, применить правило *rename_entity*(x, y, z), при получении новых ребер с учетом блокирующих доступов занести их в F^a: $F^a = F^a \cup \{(x, z, write_t)$, если $z \in E \setminus E_B\} \cup \{(x, e, write_t)$: $e \in E \setminus E_B$, $x \neq e$ и $e \le_B y\} \cup \{(x, s, write_t)$: $s \in S$, $x \neq s$ и $(s, e, \alpha_r) \in R$, где $e \in E \setminus E_B$, $e \le_B y$ и $\alpha_r \in R_r\}$.
3. Для всех ребер вида $(x, y, read_r) \in R$, $(x, y, read_a) \notin A$ применить правило *access_read*(x, y), полученные ребра $(x, y, read_a)$ и $(y, x, write_m)$ занести в A^a и F^a соответственно. Если $x \in N_S$, то $F^a = F^a \cup \{(x, e, write_t)$: $e \in E \setminus E_B$, $x \neq e$ и $y \le_B e\}$. Если $x \in L_S$, то $E'_B = E_B \cup \{y\}$ и для всех $e \in E$, таких, что $y \le_B e$, положить $H'_B(e) = \varnothing$.
4. Для всех ребер вида $(x, y, write_r) \in R$, $(x, y, write_a) \notin A$ применить правило *access_write*(x, y), полученные ребра $(x, y, write_a)$ и $(x, y, write_m)$ занести в A^a и F^a соответственно. Если $x \in N_S$, то $F^a = F^a \cup \{(x, e, write_t)$: $e \in E \setminus E_B$, $x \neq e$ и $y \le_B e\}$. Если $x \in L_S \cap S$, то $E'_B = E_B \cup \{y\}$ и для всех $e \in E$, таких, что $y \le_B e$: $H'_B(e) = \varnothing$.
5. Для всех ребер вида $(x, y, append_r) \in R$, $(x, y, append_a) \notin A$ применить правило *access_append*(x, y), полученные ребра $(x, y, append_a)$ и $(x, y, write_m)$ занести в A^a и F^a соответственно. Если $x \in N_S$, то $F^a = F^a \cup \{(x, e, write_t)$: $e \in E \setminus E_B$, $x \neq e$ и $y \le_B e\}$. Если $x \in L_S$, то $E'_B = E_B \cup \{y\}$ и для всех $e \in E$, таких, что $y \le_B e$, положить $H'_B(e) = \varnothing$.
6. $L = \{(x, y, \alpha) \in R \cup F\}$, $N = \varnothing$.
7. Выбрать из списка L первое ребро (x, y, α), занести x и y в N, удалить ребро (x, y, α) из L.
8. Для всех вершин $z, y' \in N$ проверить возможность применения правила *flow* на четверке вершин x, y, y', z с использованием ребер (x, y, α) и (z, y', β) и правил *post*, *pass*, *find* на тройке вершин x, y, z с использованием ребра (x, y, α). Новые ребра вида (a, b, γ), где $\{a, b\} \subset \{x, y, y', z\}$, $\gamma \in \{write_m, write_t\}$, получающиеся в зависимости от правила преобразования состояния БД ДП-модели и принадлежности субъектов к множествам L_S и N_S, занести в конец L и F^a.
9. Если L не пусто, перейти на шаг 6.

Теорема 7. Алгоритм 6 корректно строит *access*-замыкание графа доступов.

Доказательство теоремы выполняется аналогично доказательству теоремы 3 с учетом правил преобразования БД ДП-модели. ■

2.4. Построение замыканий графа доступов ФАС ДП-модели

Перечислим основные положения ФАС ДП-модели:

- сущность $e \in E$ называется функционально-ассоциированной с субъектом $s \in S$ в состоянии G, если данные в сущности e влияют на вид преобразования данных, реализуемого субъектом s в состоянии G;
- $[s] \subset E$ – множество всех сущностей, функционально-ассоциированных с субъектом s, при этом выполняется условие $s \in [s]$;
- только информационный поток по памяти к сущности, функционально-ассоциированной с субъектом, приводит к изменению вида преобразования данных, реализуемого этим субъектом;
- функционально-ассоциированными с субъектом являются сущности, от которых зависит вид преобразования данных, реализуемого субъектом в данном или некотором последующем состоянии системы $\Sigma(G^*,$

OP), множество сущностей, функционально-ассоциированных с субъектом, не изменяется в процессе функционирования системы;

- если субъект реализовал информационный поток по памяти от себя к сущности, функционально-ассоциированной с другим субъектом, то первый субъект получает право доступа владения ко второму субъекту (правило *control*(), табл. 4.1).

В соответствии с данными положениями модифицируются условия и результаты применения правил преобразования состояний БК ДП-модели [3, табл. 2.1].

Алгоритм 8 построения *control*-замыкания графа доступов.

1. $L = \{(y, z): z \in [y], y \in S, z \in E\}$.
2. Выбрать из списка L первую пару (y, z) . Удалить пару (y, z) из L.
3. Для всех вершин $x \in S$ применить правило *control*(x, y, z). Если в результате этого появятся новые ребра вида (x, y, own_r), то занести их в $R^{control}$.
4. Применить алгоритм 6 к графу $G^{control} = (S, E, R^{control} \cup A^{control} \cup F^{control}, H)$.
5. Если L не пусто, перейти на шаг 2.

Теорема 8. Алгоритм 8 корректно строит *control*-замыкание графа доступов.

Доказательство теоремы выполняется аналогично доказательству теоремы 3 с учетом правил преобразования ФАС ДП-модели. ■

3. Анализ путей распространения прав доступа и информационных потоков

Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ – начальное состояние системы $\Sigma(G^*, OP)$ и $\alpha \in R_{raf}$, $x, y \in E_0$, где $x \neq y$. Пусть существуют состояния $G_1, \ldots, G_N = (S_N, E_N, R_N \cup A_N \cup F_N, H_N)$ и правила преобразования состояний $op_1, \ldots, op_N$ из OP такие, что $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и $(x, y, \alpha) \in R_N \cup A_N \cup F_N$, где $N \geq 0$. Обозначим множество ребер, к которым применялись правила преобразования состояний $B \subseteq R_N \cup A_N \cup F_N$, а множество применяемых правил – OP_B. Пусть $P_{pr} \subseteq B \times OP_B$ и $P_{ps} \subseteq OP_B \times B$. *Графом анализа* системы $\Sigma(G^*, OP)$ в состоянии G_N будем называть ориентированный граф $G^A(x, y, \alpha) = (B \cup OP_B, P_{pr} \cup P_{ps})$. Условия применения правил, соответствующие условиям отношения иерархии сущностей системы, в граф анализа включать не будем.

Алгоритм 9 построения графа G^A при известной последовательности правил преобразования состояний $op_1, \ldots, op_N$.

1. $L = \{op_1, \ldots, op_N\}$, $OP_B = L$, $P_{pr} \cup P_{ps} = \varnothing$, $B = \varnothing$, $j = 0$, $j = j + 1$.
2. Выбрать из списка L первое правило op. Удалить правило op из L.
3. Если ребра $r_1, \ldots, r_m$ графов G_i для $i = 0, \ldots, j-1$ необходимы для применения правила op, а ребра $p_1, \ldots, p_l$ появляются после его применения, то $B = B \cup \{r_1, \ldots, r_m\} \cup \{p_1, \ldots, p_l\}$, $P_{pr} = P_{pr} \cup \{(r_1, op), \ldots, (r_m, op)\}$, $P_{ps} = P_{ps} \cup \{(op, p_1), \ldots, (op, p_l)\}$.
4. Если L не пусто, перейти на шаг 1.

Рассмотрим следующую задачу. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ – начальное состояние системы $\Sigma(G^*, OP)$ и $\alpha \in R_{raf}$, $x, y \in E_0$, где $x \neq y$, и пусть в некотором состоянии системы $\Sigma(G^*, OP)$ выполнено условие: $(x, y, \alpha) \in R \cup A \cup F$. Требуется найти все такие правила преобразования состояний $op_1, \ldots, op_N$, что $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и $(x, y, \alpha) \in R_N \cup A_N \cup F_N$ для некоторого $N \geq 0$. Для нахождения всех таких правил преобразования состояний ДП-моделей необходимо сначала преобразовать граф доступов в его замыкание, а затем с помощью следующих алгоритмов построить граф анализа $G^A(x, y, \alpha)$. Для краткости не будем писать элементы, добавляемые в множество B, так как их легко определить по ребрам в P_{pr}. Алгоритмы 10 – 13 строят граф $G^A(x, y, \alpha)$, который содержит всевозможные траектории получения прав доступа, доступов и реализации информационных потоков соответственно для базовой, БК, БД и ФАС ДП-моделей.

Алгоритм 10.

1. $OP_B = \varnothing$, $P_{pr} \cup P_{ps} = \varnothing$, $B = (x, y, \alpha)$.
2. Если $(x, y, \alpha) \in R_0 \cup A_0 \cup F_0$, то выход.
3. Если $\alpha \in R_r$:
   - 3.1. Если $\alpha = own_r$, существует $z \in E \setminus S$, что $y \in H(z)$, $(x, z, \beta) \in R$, $\beta \in \{write_r, append_r\}$, и для всех $e \in E$, что $y \leq e$, выполнено $(x, e, write_l) \in F$, то $OP_B = OP_B \cup create\_entity(x, y, z)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), create\_entity(x, y, z))$, $P_{ps} = P_{ps} \cup (create\_entity(x, y, z), (x, y, \alpha))$.
   - 3.2. Если $\alpha = own_r$, $y \in S$, $y \in H(x)$, $(y, x, write_l) \in F$, то для всех $z \in E$, что $(x, z, execute_r) \in R$, и для всех $e \in E$, что $z \leq e$, выполнено $(x, e, write_l) \in F$, положить $OP_B = OP_B \cup create\_subject(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_entity(x, z, y))$, $P_{ps} = P_{ps} \cup (create\_entity(x, z, y), (x, y, \alpha))$.
   - 3.3. Если $\alpha \neq own_r$ и $(x, y, own_r) \in R$, то $OP_B = OP_B \cup own\_take(\alpha, x, y)$, $P_{pr} = P_{pr} \cup ((x, y, own_r), own\_take(\alpha, x, y))$, $P_{ps} = P_{ps} \cup (own\_take(\alpha, x, y), (x, y, \alpha))$.
   - 3.4. Для всех $z \in S$, таких, что (x, z, own_r), $(z, y, \alpha) \in R$, положить $OP_B = OP_B \cup take\_right(\alpha, x, z, y)$, $P_{pr} = P_{pr} \cup ((x,z,own_r), take\_right(\alpha,x,z,y)) \cup ((z, y, \alpha), take\_right(\alpha, x, z, y))$, $P_{ps} = P_{ps} \cup (take\_right(\alpha, x, z, y), (x, y, \alpha))$.

3.5. Для всех $z \in S$, таких, что (z, x, own_r), $(z, y, \alpha) \in R$, положить $OP_B = OP_B \cup grant\_right(\alpha, z, x, y)$, $P_{pr} = P_{pr} \cup ((z, y, own_r), grant\_right(\alpha, z, x, y)) \cup ((z, y, \alpha), grant\_right(\alpha, z, x, y))$, $P_{ps} = P_{ps} \cup (grant\_right(\alpha, z, x, y), (x, y, \alpha))$.

4. Если $\alpha \in R_a$:
   4.1. Если $\alpha = read_a$ и $(x, y, read_r) \in R$, то $OP_B = OP_B \cup access\_read(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, read_r), access\_read(x, y))$, $P_{ps} = P_{ps} \cup (access\_read(x, y), (x, y, \alpha))$.
   4.2. Если $\alpha = write_a$ и $(x, y, write_r) \in R$, то $OP_B = OP_B \cup access\_write(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, write_r), access\_write(x, y))$, $P_{ps} = P_{ps} \cup (access\_write(x, y), (x, y, \alpha))$.
   4.3. Если $\alpha = append_a$ и $(x, y, append_r) \in R$, то $OP_B = OP_B \cup access\_append(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, append_r), access\_append(x, y))$, $P_{ps} = P_{ps} \cup (access\_append(x, y), (x, y, \alpha))$.
5. Если $\alpha = write_m$:
   5.1. Если $(x, y, read_a) \in A$, то $OP_B = OP_B \cup access\_read(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, read_r), access\_read(x, y))$, $P_{ps} = P_{ps} \cup (access\_read(x, y), (x, y, \alpha))$.
   5.2. Если $(x, y, write_a) \in A$, то $OP_B = OP_B \cup access\_write(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, write_r), access\_write(x, y))$, $P_{ps} = P_{ps} \cup (access\_write(x, y), (x, y, \alpha))$.
   5.3. Если $(x, y, append_a) \in A$, то $OP_B = OP_B \cup access\_append(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, append_r), access\_append(x, y))$, $P_{ps} = P_{ps} \cup (access\_append(x, y), (x, y, \alpha))$.
   5.4. Для всех $z \in S$, таких, что (x, z, β), $(z, y, \gamma) \in R \cup F$ и $\beta, \gamma \in \{write_r, append_r, write_m\}$, положить $OP_B = OP_B \cup find(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), find(x, z, y)) \cup ((z, y, \gamma), find(x, z, y))$, $P_{ps} = P_{ps} \cup (find(x, z, y), (x, y, \alpha))$.
   5.5. Для всех $z \in E$, таких, что (x, z, β), $(y, z, read_r) \in R \cup F$ и $\beta \in \{write_r, append_r, write_m\}$, положить $OP_B = OP_B \cup post(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), post(x, z, y)) \cup ((y, z, read_r), post(x, z, y))$, $P_{ps} = P_{ps} \cup (post(x, z, y), (x, y, \alpha))$.
   5.6. Для всех $z \in S$, таких, что $(z, x, read_r) \in R \cup F$, $(z, y, \beta) \in R \cup F$ и $\beta \in \{write_r, append_r, write_m\}$, положить $OP_B = OP_B \cup pass(x, z, y)$, $P_{pr} = P_{pr} \cup ((z, x, read_r), pass(x, z, y)) \cup ((z, y, \beta), pass(x, z, y))$, $P_{ps} = P_{ps} \cup (pass(x, z, y), (x, y, \alpha))$.
6. Если $\alpha = write_t$:
   6.1. Для всех $z \in E$, таких, что $(x, z, read_a) \in A$ и $z \le y$, положить $OP_B = OP_B \cup access\_read(x, z)$, $P_{pr} = P_{pr} \cup ((x, z, read_r), access\_read(x, z))$, $P_{ps} = P_{ps} \cup (access\_read(x, z), (x, y, \alpha))$.
   6.2. Для всех $z \in E$, таких, что $(x, z, write_a) \in A$ и $z \le y$, положить $OP_B = OP_B \cup access\_write(x, z)$, $P_{pr} = P_{pr} \cup ((x, z, write_r), access\_write(x, z))$, $P_{ps} = P_{ps} \cup (access\_write(x, z), (x, y, \alpha))$.
   6.3. Для всех $z \in E$, таких, что $(x, z, append_a) \in A$ и $z \le y$, положить $OP_B = OP_B \cup access\_append(x, z)$, $P_{pr} = P_{pr} \cup ((x, z, append_r), access\_append(x, z))$, $P_{ps} = P_{ps} \cup (access\_append(x, z), (x, y, \alpha))$.
   6.4. Если $(y, x, own_r) \in R$, то для всех $z \in E$, таких, что $R_x \cap R_y \neq \varnothing$, где $R_y = \{\beta: (y, z, \beta) \in R\}$, $R_x = \{\beta: (x, z, \beta) \in R\}$, положить $OP_B = OP_B \cup take\_right(y, x, z)$, $P_{pr} = P_{pr} \cup ((y, x, own_r), take\_right(y, x, z))$, $P_{ps} = P_{ps} \cup (take\_right(y, x, z), (x, y, \alpha))$.
   6.5. Если $(x, y, own_r) \in R$, то для всех $z \in E$, таких, что $R_x \cap R_y \neq \varnothing$, где $R_y = \{\beta: (y, z, \beta) \in R\}$, $R_x = \{\beta: (x, z, \beta) \in R\}$, положить $OP_B = OP_B \cup grant\_right(x, y, z)$, $P_{pr} = P_{pr} \cup ((x, y, own_r), grant\_right(x, y, z))$, $P_{ps} = P_{ps} \cup (grant\_right(x, y, z), (x, y, \alpha))$.
   6.6. Если $(x, y, write_r) \in R$, то для всех $z \in H(y)$, таких, что $\{(x, e, write_t): e \in E, x \neq e$ и $e \le z\} \cup \{(x, s, write_t): s \in S, x \neq s$ и $(s, e, \alpha_r) \in R$, где $e \in E, e \le z$ и $\alpha_r \in R_r\} \subseteq F$, положить $OP_B = OP_B \cup rename\_entity(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, y, write_r), rename\_entity(x, z, y))$, $P_{ps} = P_{ps} \cup (rename\_entity(x, z, y), (x, y, \alpha))$.
   6.7. Для всех $e \in E$, таких, что $y \le e$, $e \in H(z)$, $(x, z, write_r) \cup (x, z, write_t) \cup \{(x, s, write_t): s \in S, x \neq s$ и $(s, e', \alpha_r) \in R$, где $e' \in E, e' \le e$ и $\alpha_r \in R_r\} \subseteq R \cup F$, положить $OP_B = OP_B \cup rename\_entity(x, e, z)$, $P_{pr} = P_{pr} \cup ((x, z, write_r), rename\_entity(x, e, z))$, $P_{ps} = P_{ps} \cup (rename\_entity(x, e, z), (x, y, \alpha))$.
   6.8. Если $y \in S$, то для всех $z, z' \in E$, таких, что $z \in H(z')$, $(x, z', write_r)$, $(x, z', write_t) \in R \cup F$, если $e \le z$ и $(y, e, \alpha_r) \in R$, где $e \in E$, $\alpha_r \in R_r$, то положить $OP_B = OP_B \cup rename\_entity(x, z, z')$, $P_{pr} = P_{pr} \cup ((x, z', write_r), rename\_entity(x, z, z')) \cup ((y, e, \alpha_r), rename\_entity(x, z, z'))$, $P_{ps} = P_{ps} \cup (rename\_entity(x, z, z'), (x, y, \alpha))$.
   6.9. Для всех $z, z' \in E$, что $z \in H(z')$, $(x, z, own_r) \in R$, $(x, z', \beta) \in R$, $\beta \in \{write_r, append_r\}$ и $z \le y$, положить $OP_B = OP_B \cup create\_entity(x, z', z)$, $P_{pr} = P_{pr} \cup ((x, y, \beta), create\_entity(x, z', z))$, $P_{ps} = P_{ps} \cup (create\_entity(x, z', z), (x, y, \alpha))$.
   6.10. Для всех $z, z' \in E$, что $(x, z, execute_r) \in R$, $z' \in H(x)$, $(x, z', own_r) \in R$ и $z \le y$, положить $OP_B = OP_B \cup create\_subjecty(x, z, z')$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_subject(x, z, z'))$, $P_{ps} = P_{ps} \cup (create\_subject(x, z, z'), (x, y, \alpha))$.
   6.11. Если $y \in H(x)$, $(x, y, own_r) \in R$, то для всех $z \in E$, что $(x, z, execute_r) \in R$, положить $OP_B = OP_B \cup create\_subjecty(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_subject(x, z, y))$, $P_{ps} = P_{ps} \cup (create\_subject(x, z, y), (x, y, \alpha))$.

6.12. Если $y \in S$, $(y, x, write_t) \in F$, то для всех $z, z' \in E$, что $z \le z'$ (x, z, α_r), $(y, z', \beta_r) \in R$, где $\alpha_r, \beta_r \in R_r$, положить $OP_B = OP_B \cup flow(x, z, z', y)$, $P_{pr} = P_{pr} \cup ((x, z, \alpha_r), flow(x, z, z', y)) \cup ((y, z', \beta_r), flow(x, z, z', y))$, $P_{ps} = P_{ps} \cup (flow(x, z, z', y), (x, y, \alpha))$.

6.13. Если $y \in S$, $(y, x, write_t) \in F$, то для всех $z, z' \in E$, что $z \le z'$ (y, z, α_r), $(x, z', \beta_r) \in R$, где $\alpha_r, \beta_r \in R_r$, положить $OP_B = OP_B \cup flow(x, z, z', y)$, $P_{pr} = P_{pr} \cup ((x, z', \alpha_r), flow(x, z, z', y)) \cup ((y, z, \beta_r), flow(x, z, z', y))$, $P_{ps} = P_{ps} \cup (flow(x, z, z', y), (x, y, \alpha))$.

6.14. Для всех $z \in S$, если (x, z, γ), $(z, y, \beta) \in F$, $write_t \in \{\gamma, \beta\}$, то $OP_B = OP_B \cup find(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \gamma), find(x, z, y)) \cup ((z, y, \beta), find(x, z, y))$, $P_{ps} = P_{ps} \cup (find(x, z, y), (x, y, \alpha))$.

6.15. Для всех $z \in E$, если $(x, z, write_t)$, $(y, z, read_r) \in R \cup F$, то $OP_B = OP_B \cup post(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, write_t), post(x, z, y)) \cup ((y, z, read_r), post(x, z, y))$, $P_{ps} = P_{ps} \cup (post(x, z, y), (x, y, \alpha))$.

6.16. Для всех $z \in S$, если $(z, x, read_r)$, $(z, y, \beta) \in R \cup F$, где $\beta \in \{write_r, append_r, write_m, write_t\}$, то $OP_B = OP_B \cup pass(x, z, y)$, $P_{pr} = P_{pr} \cup ((z, x, read_r), pass(x, z, y)) \cup ((z, y, \beta), pass(x, z, y))$, $P_{ps} = P_{ps} \cup (pass(x, z, y), (x, y, \alpha))$.

7. Для всех элементов вида $((a, b, \beta), op)$, добавленных на шагах 3 – 6 алгоритма в множество P_{pr} выполнить шаги 2–7 алгоритма 10 для (x, y, α), где $x = a$, $y = b$, $\alpha = \beta$.

Алгоритм 11 построения графа анализа $G^A(x, y, \alpha)$ для БК ДП-модели.

1. $OP_B = \emptyset$, $P_{pr} \cup P_{ps} = \emptyset$, $B = (x, y, \alpha)$.
2. Если $(x, y, \alpha) \in R_0 \cup A_0 \cup F_0$, то выход.
3. Если $x \in N_S$, выполнить шаги 2 – 6 алгоритма 10.
4. Если $x \in L_S$, $\alpha \in R_r$:
   4.1. Если $\alpha = own_r$ и существует $z \in E \backslash S$, что $y \in H(z)$, $(x, z, \beta) \in R$, $\beta \in \{write_r, append_r\}$, то $OP_B = OP_B \cup create\_entity(x, y, z)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), create\_entity(x, y, z))$, $P_{ps} = P_{ps} \cup (create\_entity(x, y, z), (x, y, \alpha))$.
   4.2. Если $\alpha = own_r$, $y \in S$, $y \in H(x)$, $(y, x, write_t) \in F$, то для всех $z \in E$, что $(x, z, execute_r) \in R$, положить $OP_B = OP_B \cup create\_subject(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_entity(x, z, y))$, $P_{ps} = P_{ps} \cup (create\_entity(x, z, y), (x, y, \alpha))$.
   4.3. Если $\alpha \ne own_r$, то $OP_B = OP_B \cup own\_take(\alpha, x, y)$, $P_{pr} = P_{pr} \cup ((x, y, own_r), own\_take(\alpha, x, y))$, $P_{ps} = P_{ps} \cup (own\_take(\alpha, x, y), (x, y, \alpha))$.
5. Если $x \in L_S$, $\alpha \in R_a$:
   5.1. Если $\alpha = read_a$ и $(x, y, read_r) \in R$, то $OP_B = OP_B \cup access\_read(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, read_r), access\_read(x, y))$, $P_{ps} = P_{ps} \cup (access\_read(x, y), (x, y, \alpha))$.
   5.2. Если $\alpha = write_a$ и $(x, y, write_r) \in R$, то $OP_B = OP_B \cup access\_write(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, write_r), access\_write(x, y))$, $P_{ps} = P_{ps} \cup (access\_write(x, y), (x, y, \alpha))$.
   5.3. Если $\alpha = append_a$ и $(x, y, append_r) \in R$, то $OP_B = OP_B \cup access\_append(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, append_r), access\_append(x, y))$, $P_{ps} = P_{ps} \cup (access\_append(x, y), (x, y, \alpha))$.
6. Если $x \in L_S$, если $\alpha = write_m$:
   6.1. Если $(x, y, read_a) \in A$, то $OP_B = OP_B \cup access\_read(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, read_r), access\_read(x, y))$, $P_{ps} = P_{ps} \cup (access\_read(x, y), (x, y, \alpha))$.
   6.2. Если $(x, y, write_a) \in A$, то $OP_B = OP_B \cup access\_write(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, write_r), access\_write(x, y))$, $P_{ps} = P_{ps} \cup (access\_write(x, y), (x, y, \alpha))$.
   6.3. Если $(x, y, append_a) \in A$, то $OP_B = OP_B \cup access\_append(x, y)$, $P_{pr} = P_{pr} \cup ((x, y, append_r), access\_append(x, y))$, $P_{ps} = P_{ps} \cup (access\_append(x, y), (x, y, \alpha))$.
   6.4. Для всех $z \in S$, таких, что (x, z, β), $(z, y, \gamma) \in R \cup F$ и $\beta, \gamma \in \{write_r, append_r, write_m\}$, положить $OP_B = OP_B \cup find(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), find(x, z, y)) \cup ((z, y, \gamma), find(x, z, y))$, $P_{ps} = P_{ps} \cup (find(x, z, y), (x, y, \alpha))$.
   6.5. Для всех $z \in E$, таких, что (x, z, β), $(y, z, read_r) \in R \cup F$ и $\beta \in \{write_r, append_r, write_m\}$, положить $OP_B = OP_B \cup post(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), post(x, z, y)) \cup ((y, z, read_r), post(x, z, y))$, $P_{ps} = P_{ps} \cup (post(x, z, y), (x, y, \alpha))$.
   6.6. Для всех $z \in S$, таких, что $(z, x, read_r) \in R \cup F$, $(z, y, \beta) \in R \cup F$ и $\beta \in \{write_r, append_r, write_m\}$, положить $OP_B = OP_B \cup pass(x, z, y)$, $P_{pr} = P_{pr} \cup ((z, x, read_r), pass(x, z, y)) \cup ((z, y, \beta), pass(x, z, y))$, $P_{ps} = P_{ps} \cup (pass(x, z, y), (x, y, \alpha))$.
7. Для всех элементов вида $((a, b, \beta), op)$, добавленных на шагах 3 – 6 алгоритма 11 в множество P_{pr}, выполнить шаги 2–7 алгоритма 11 для (x, y, α), где $x = a$, $y = b$, $\alpha = \beta$.

Алгоритм 12 построения графа анализа $G^A(x, y, \alpha)$ для БК ДП-модели.

1. $OP_B = \emptyset$, $P_{pr} \cup P_{ps} = \emptyset$, $B = (x, y, \alpha)$.

2. Если $(x, y, \alpha) \in R_0 \cup A_0 \cup F_0$, то выход.
3. Если $x \in L_S$, $\alpha \in R_r$:
   3.1. Если $\alpha = own_r$ и существует $z \in E \setminus S$, что $y \in H_B(z)$, $(x, z, \beta) \in R$, $\beta \in \{write_r, append_r\}$, то $OP_B = OP_B \cup create\_entity(x, y, z)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), create\_entity(x, y, z))$, $P_{ps} = P_{ps} \cup (create\_entity(x, y, z), (x, y, \alpha))$.
   3.2. Если $\alpha = own_r$, $y \in S$, $y \in H_B(x)$, то для всех $z \in E$, что $(x, z, execute_r) \in R$, положить $OP_B = OP_B \cup create\_subject(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_entity(x, z, y))$, $P_{ps} = P_{ps} \cup (create\_entity(x, z, y), (x, y, \alpha))$.
   3.3. Если $\alpha \neq own_r$, то $OP_B = OP_B \cup own\_take(\alpha, x, y)$, $P_{pr} = P_{pr} \cup ((x, y, own_r), own\_take(\alpha, x, y))$, $P_{ps} = P_{ps} \cup (own\_take(\alpha, x, y), (x, y, \alpha))$.
4. Если $x \in L_S$, $\alpha \in R_a$, выполнить шаг 5 алгоритма 11.
5. Если $x \in L_S$, если $\alpha = write_m$:
   5.1. Выполнить шаги 6.1 – 6.3 алгоритма 11.
   5.2. Для всех $z \in S$, таких, что $(x, z, write_m)$, $(z, y, write_m) \in F$, положить $OP_B = OP_B \cup find(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, write_m), find(x, z, y)) \cup ((z, y, write_m), find(x, z, y))$, $P_{ps} = P_{ps} \cup (find(x, z, y), (x, y, \alpha))$.
   5.3. Для всех $z \in E$, таких, что $(x, z, write_m)$, $(y, z, read_a) \in A$, положить $OP_B = OP_B \cup post(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), post(x, z, y)) \cup ((y, z, read_a), post(x, z, y))$, $P_{ps} = P_{ps} \cup (post(x, z, y), (x, y, \alpha))$.
   5.4. Для всех $z \in S$, таких, что $(z, x, read_a) \in A$, $(z, y, write_m) \in F$, положить $OP_B = OP_B \cup pass(x, z, y)$, $P_{pr} = P_{pr} \cup ((z, x, read_a), pass(x, z, y)) \cup ((z, y, write_m), pass(x, z, y))$, $P_{ps} = P_{ps} \cup (pass(x, z, y), (x, y, \alpha))$.
6. Если $x \in N_S$, $\alpha \in R_r$:
   6.1. Если $\alpha = own_r$, существует $z \in E \setminus S$, что $y \in H_B(z)$, $(x, z, \beta) \in R$, $\beta \in \{write_r, append_r\}$, и для всех $e \in E \setminus E_B$, что $y \leq_B e$, выполнено $(x, e, write_t) \in F$, то $OP_B = OP_B \cup create\_entity(x, y, z)$, $P_{pr} = P_{pr} \cup ((x, z, \beta), create\_entity(x, y, z))$, $P_{ps} = P_{ps} \cup (create\_entity(x, y, z), (x, y, \alpha))$.
   6.2. Если $\alpha = own_r$, $y \in S$, $y \in H_B(x)$, $(y, x, write_t) \in F$, то для всех $z \in E$, что $(x, z, execute_r) \in R$, и для всех $e \in E \setminus E_B$, что $z \leq_B e$, выполнено $(x, e, write_t) \in F$, положить $OP_B = OP_B \cup create\_subject(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_entity(x, z, y))$, $P_{ps} = P_{ps} \cup (create\_entity(x, z, y), (x, y, \alpha))$.
   6.3. Выполнить шаги 3.3 – 3.5 алгоритма 10.
7. Выполнить шаг 4 алгоритма 10.
8. Если $x \in N_S$, $\alpha = write_m$, выполнить шаги 5.1 – 5.3 алгоритма 10 и 5.2 – 5.4 алгоритма 12.
9. Если $x \in N_S$, $\alpha = write_t$:
   9.1. Для всех $z \in E$, таких, что $(x, z, read_a) \in A$ и $y \in E \setminus E_B$, $x \neq y$, $z \leq_B y$, положить $OP_B = OP_B \cup access\_read(x, z)$, $P_{pr} = P_{pr} \cup ((x, z, read_r), access\_read(x, z))$, $P_{ps} = P_{ps} \cup (access\_read(x, z), (x, y, \alpha))$.
   9.2. Для всех $z \in E$, таких, что $(x, z, write_a) \in A$ и $y \in E \setminus E_B$, $x \neq y$, $z \leq_B y$, положить $OP_B = OP_B \cup access\_write(x, z)$, $P_{pr} = P_{pr} \cup ((x, z, write_r), access\_write(x, z))$, $P_{ps} = P_{ps} \cup (access\_write(x, z), (x, y, \alpha))$.
   9.3. Для всех $z \in E$, таких, что $(x, z, append_a) \in A$ и $y \in E \setminus E_B$, $x \neq y$, $z \leq_B y$, положить $OP_B = OP_B \cup access\_append(x, z)$, $P_{pr} = P_{pr} \cup ((x, z, append_r), access\_append(x, z))$, $P_{ps} = P_{ps} \cup (access\_append(x, z), (x, y, \alpha))$.
   9.4. Выполнить шаги 6.4 – 6.5 алгоритма 10.
   9.5. Если $(x, y, write_r) \in R$, то для всех $z \in H(y)$, таких, что $\{(x, e, write_t)$: $e \in E \setminus E_B$, $x \neq e$ и $e \leq_B z\} \cup \{(x, s, write_t)$: $s \in S$, $x \neq s$ и $(s, e, \alpha_r) \in R$, где $e \in E \setminus E_B$, $e \leq_B z$ и $\alpha_r \in R_r\} \subseteq F$, положить $OP_B = OP_B \cup rename\_entity(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, y, write_r), rename\_entity(x, z, y))$, $P_{ps} = P_{ps} \cup (rename\_entity(x, z, y), (x, y, \alpha))$.
   9.6. Для всех $e \in E$, таких, что $y \leq_B e$, $e \in H(z)$, $(x, z, write_r) \cup (x, z, write_t) \cup \{(x, s, write_t)$: $s \in S$, $x \neq s$ и $(s, e', \alpha_r) \in R$, где $e' \in E \setminus E_B$, $e' \leq_B e$ и $\alpha_r \in R_r\} \subseteq R \cup F$, положить $OP_B = OP_B \cup rename\_entity(x, e, z)$, $P_{pr} = P_{pr} \cup ((x, z, write_r), rename\_entity(x, e, z))$, $P_{ps} = P_{ps} \cup (rename\_entity(x, e, z), (x, y, \alpha))$.
   9.7. Если $y \in S$, то для всех $z, z' \in E$, таких, что $z \in H(z')$, $(x, z', write_r)$, $(x, z', write_t) \in R \cup F$, если $e \leq_B z$ и $(y, e, \alpha_r) \in R$, где $e \in E \setminus E_B$, $\alpha_r \in R_r$, то положить $OP_B = OP_B \cup rename\_entity(x, z, z')$, $P_{pr} = P_{pr} \cup ((x, z', write_r), rename\_entity(x, z, z')) \cup ((y, e, \alpha_r), rename\_entity(x, z, z'))$, $P_{ps} = P_{ps} \cup (rename\_entity(x, z, z'), (x, y, \alpha))$.
   9.8. Для всех $z, z' \in E$, что $z \in H_B(z')$, $(x, z, own_r) \in R$, $(x, z', \beta) \in R$, $\beta \in \{write_r, append_r\}$ и $z \leq_B y$, положить $OP_B = OP_B \cup create\_entity(x, z', z)$, $P_{pr} = P_{pr} \cup ((x, y, \beta), create\_entity(x, z', z))$, $P_{ps} = P_{ps} \cup (create\_entity(x, z', z), (x, y, \alpha))$.
   9.9. Для всех $z, z' \in E$, что $(x, z, execute_r) \in R$, $z' \in H(x)$, $(x, z', own_r) \in R$ и $z \leq_B y$, положить $OP_B = OP_B \cup create\_subjecty(x, z, z')$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_subject(x, z, z'))$, $P_{ps} = P_{ps} \cup (create\_subject(x, z, z'), (x, y, \alpha))$.
   9.10. Если $y \in H_B(x)$, $(x, y, own_r) \in R$, то для всех $z \in E$, что $(x, z, execute_r) \in R$, положить $OP_B = OP_B \cup create\_subjecty(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, execute_r), create\_subject(x, z, y))$, $P_{ps} = P_{ps} \cup (create\_subject(x, z, y), (x, y, \alpha))$.

9.11. Если $y \in S$, $(y, x, write_t) \in F$, то для всех $z, z' \in E$, что $z \le_B z'$, (x, z, α_r), $(y, z', \beta_r) \in R$, где $\alpha_r, \beta_r \in R_r$, положить $OP_B = OP_B \cup flow(x, z, z', y)$, $P_{pr} = P_{pr} \cup ((x, z, \alpha_r), flow(x, z, z', y)) \cup ((y, z', \beta_r), flow(x, z, z', y))$, $P_{ps} = P_{ps} \cup (flow(x, z, z', y), (x, y, \alpha))$.

9.12. Если $y \in S$, $(y, x, write_t) \in F$, то для всех $z, z' \in E$, что $z \le_B z'$, (y, z, α_r), $(x, z', \beta_r) \in R$, где $\alpha_r, \beta_r \in R_r$, положить $OP_B = OP_B \cup flow(x, z, z', y)$, $P_{pr} = P_{pr} \cup ((x, z', \alpha_r), flow(x, z, z', y)) \cup ((y, z, \beta_r), flow(x, z, z', y))$, $P_{ps} = P_{ps} \cup (flow(x, z, z', y), (x, y, \alpha))$.

9.13. Для всех $z \in S$, если (x, z, γ), $(z, y, \beta) \in F$, $\gamma, \beta \in \{write_t, write_m\}$, то $OP_B = OP_B \cup find(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, \gamma), find(x, z, y)) \cup ((z, y, \beta), find(x, z, y))$, $P_{ps} = P_{ps} \cup (find(x, z, y), (x, y, \alpha))$.

9.14. Для всех $z \in E$, если $(x, z, write_t)$, $(y, z, read_a) \in A \cup F$, то $OP_B = OP_B \cup post(x, z, y)$, $P_{pr} = P_{pr} \cup ((x, z, write_t), post(x, z, y)) \cup ((y, z, read_r), post(x, z, y))$, $P_{ps} = P_{ps} \cup (post(x, z, y), (x, y, \alpha))$.

9.15. Для всех $z \in S$, если $(z, x, read_a)$, $(z, y, \beta) \in A \cup F$, где $\beta \in \{write_r, append_r, write_m, write_t\}$, то $OP_B = OP_B \cup pass(x, z, y)$, $P_{pr} = P_{pr} \cup ((z, x, read_r), pass(x, z, y)) \cup ((z, y, \beta), pass(x, z, y))$, $P_{ps} = P_{ps} \cup (pass(x, z, y), (x, y, \alpha))$.

10. Для всех элементов вида $((a, b, \beta), op)$, добавленных на шагах 3 – 6 алгоритма 12 в множество P_{pr}, выполнить шаги 2 – 10 алгоритма 12 для (x, y, α), где $x = a$, $y = b$, $\alpha = \beta$.

Для построения графа анализа $G^A(x, y, \alpha)$ ФАС ДП-модели модифицируем алгоритм 11 построения графа анализа БК ДП-модели.

Алгоритм 13 построения графа анализа $G^A(x, y, \alpha)$ ФАС ДП-модели получается из алгоритма 11 добавлением в него следующего шага, выполняемого после шага 2:

3. Если $\alpha = own_r$, то для всех $z \in E$, $z \in [y]$, что $(x, z, write_m) \in R$, положить $OP_B = OP_B \cup control(x, y, z)$, $P_{pr} = P_{pr} \cup ((x, z, write_m), control(x, y, z))$, $P_{ps} = P_{ps} \cup (control(x, y, z), (x, y, \alpha))$.

4. Пример построения графа анализа ФАС ДП-модели

Рассмотрим сеть со следующей конфигурацией (рис. 2):

- на узле *Fw* установлена уязвимая версия *ssh*-службы, позволяющая получить права пользователя *root* через переполнение буфера;
- на узле *Ws* установлена уязвимая версия *web*-сервера, запущенного с привилегиями пользователя *apache*;
- пользователь *apache* имеет право доступа на чтение к базе данных *db*;
- в начальном состоянии нарушитель *A* на узле *Attacker* имеет доступ только к службе *ssh*, запущенной на *Fw*.

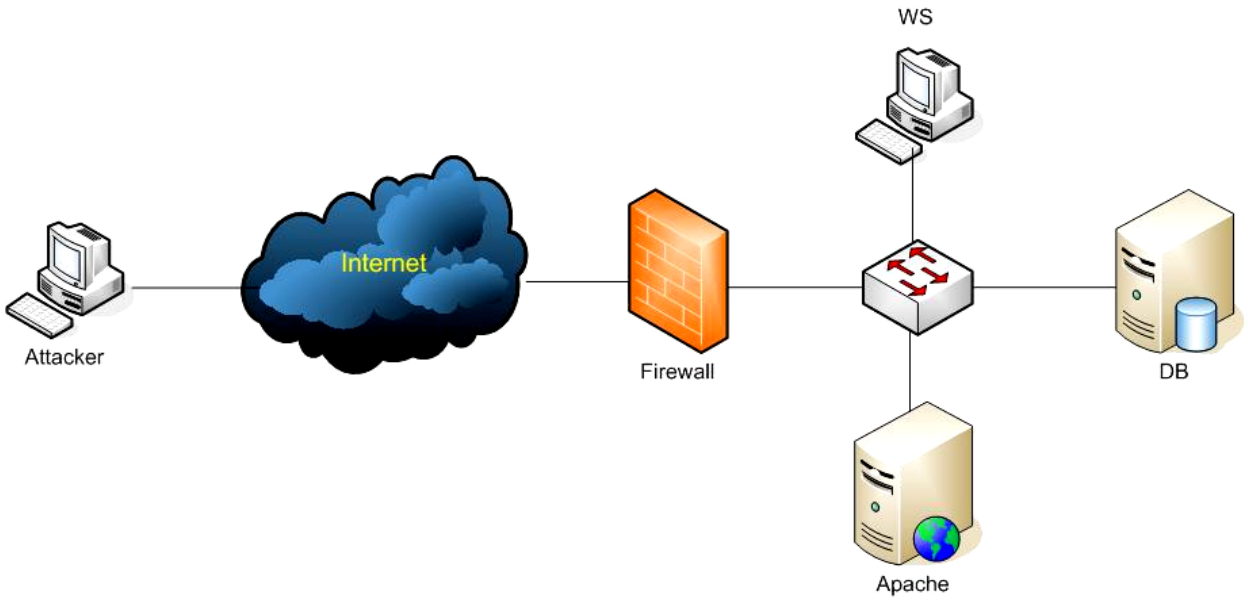


Рис. 2

На рис. 3 показана последовательность правил преобразований, в результате применения которой возможна утечка права доступа $(A, sw, read_r)$, после чего нарушитель может, используя уязвимость web-сервера, получить доступ к базе данных *db*. Далее строится граф анализа $G^A(A, sw, read_r)$ с помощью алгоритма 13. Результат представлен на рис. 4.

Примечание. Построение замыкания графа доступов на рисунках не изображено.

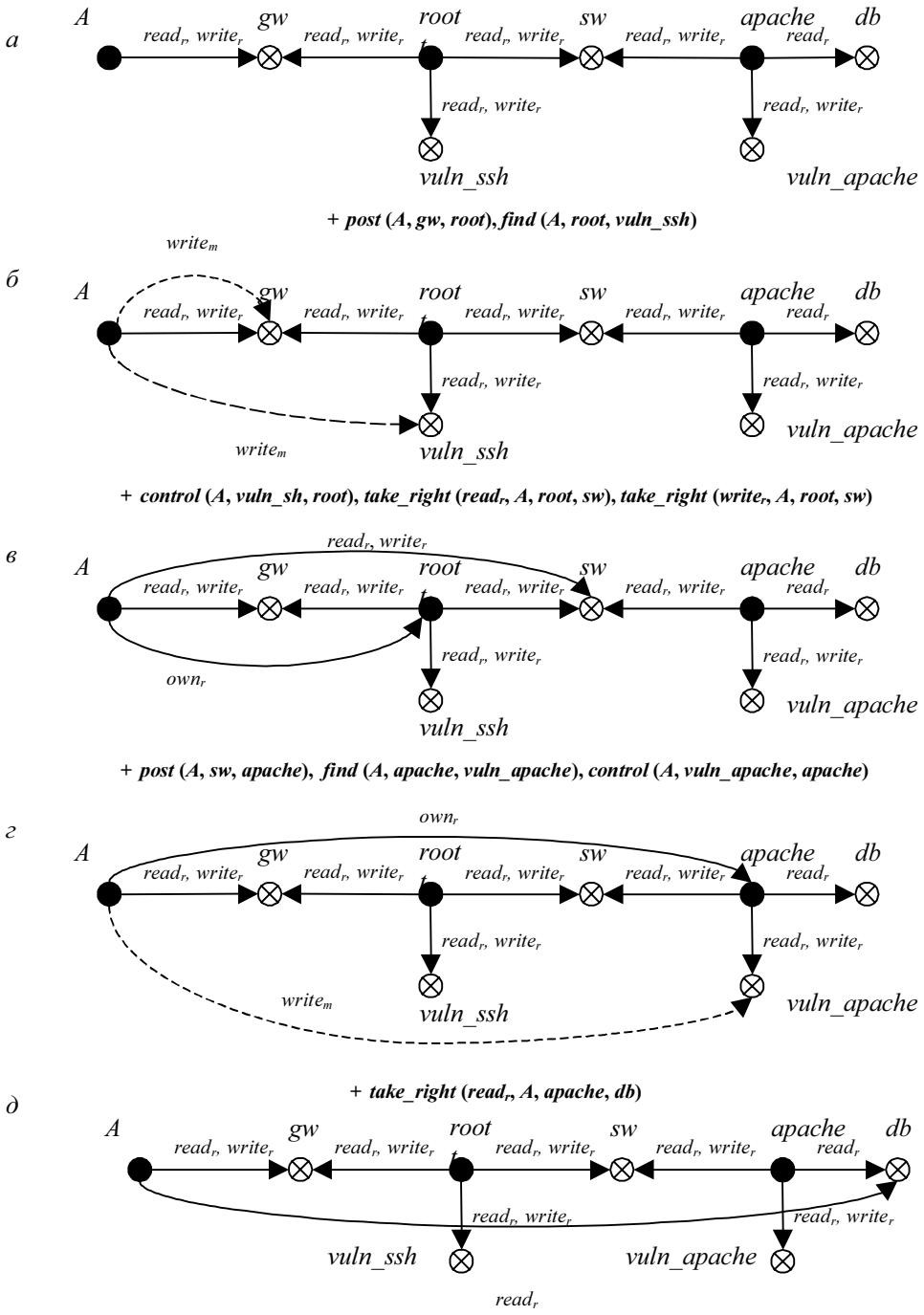


Рис. 3

5. Предотвращение утечки прав доступа и реализации запрещенных информационных потоков путем управления правами доступами в начальном состоянии ДП-модели

Пусть в системе $\Sigma(G^*, OP, G_0)$ возможна утечка права доступа (x, y, α) или реализация запрещенного информационного потока (x', y', α'). Тогда по определению возможен переход системы в состояние $G' = (S', E', R' \cup A' \cup F', H')$, в котором $(x, y, \alpha) \in R'$, $(x', y', \alpha') \in F'$, где $x \in S_0$, $x', y', y \in E_0$, $x \neq y$, $x' \neq y'$, $\alpha \in R_r$, $\alpha' \in R_f$ и $(x, y, \alpha) \notin R_0$, $(x', y', \alpha') \notin F_0$.

Формализуем и применим идею метода рекурсивного алгебраического анализа графов атак [6, 7] для предотвращения утечки прав доступа и реализации запрещенных информационных потоков.

Идея метода заключается в следующем. Правила преобразования, права доступа, доступы и информационные потоки интерпретируются логическими переменными. Взаимосвязь между ними характеризуется логическим выражением. Над логическими переменными, соответствующими правам доступа и информационным потокам исходного состояния некоторого правила преобразования, выполняется операция конъ-

юнкция. Над логическими переменными, соответствующими правилам преобразования, выполняется операция дизъюнкция, так как получение права доступа или реализация информационного потока возможно в результате применения нескольких правил преобразования. Такая последовательность действий производится рекурсивно, пока не будут достигнуты права доступа начального состояния. В результате проделанных действий будем иметь ДНФ некоторой булевой функции. Если ее приравнять к 0, найти все решения, то исключение прав доступа, соответствующих найденным решениям из множества R_0, позволяет предотвратить утечки прав доступа и реализации запрещенных информационных потоков.

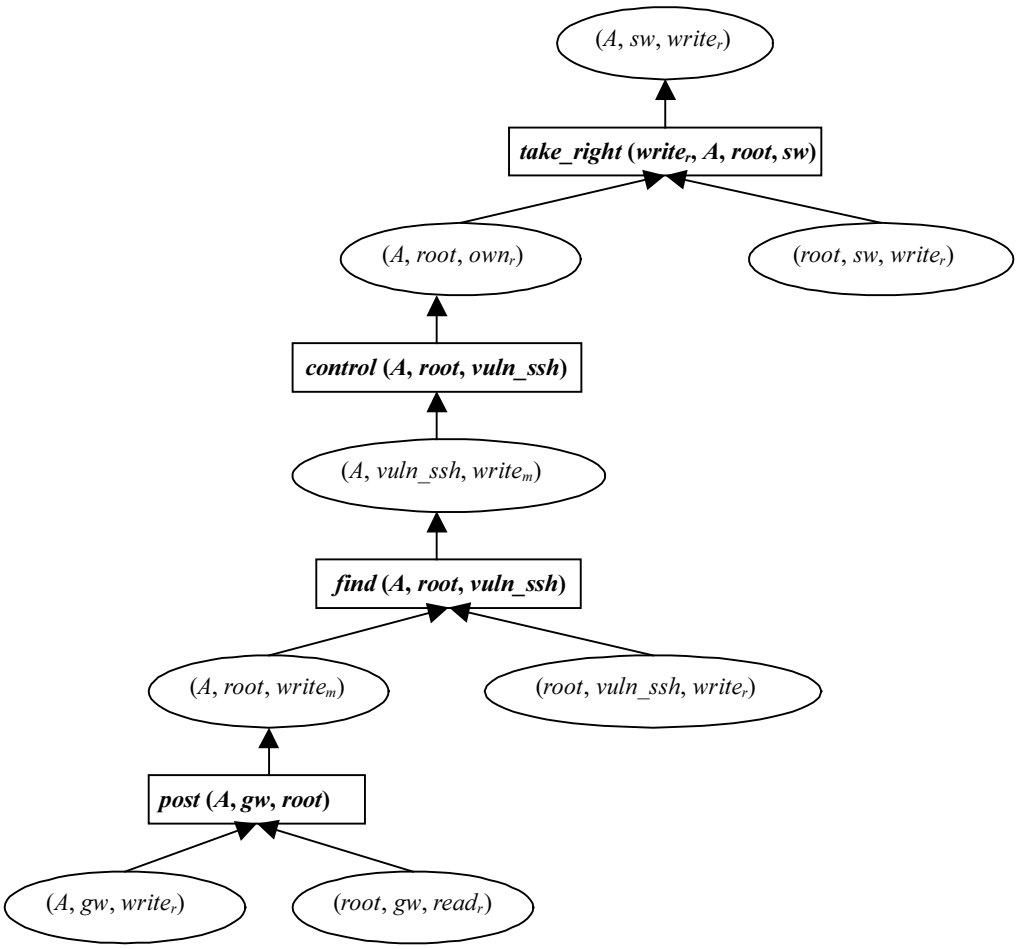


Рис. 4

Алгоритм 14.

1. Построить замыкание графа доступов ДП-модели и граф анализа $G^A(x, y, \alpha)$, вершину (x, y, α) обозначить c, положить $D = c$.
2. Ребра (op_1, c), …, (op_m, c) обозначить через $d_1,\ldots, d_m$, $L = \varnothing$.
3. Положить $c = d_1 \vee \ldots \vee d_m$.
4. Для всех $i = 1, \ldots, m$, ребра вида $((a_j, b_j, \alpha_j), op_i)$ обозначить c_{ij}, (a_j, b_j, α_j) добавить в L.
5. Каждое d_i в дизъюнкции c заменить на конъюнкцию $c_{i1}\, c_{i2\ldots}c_{in}$.
6. Изменить c в D.
7. Выполнить шаги 2–7 для всех $c \in L \backslash R_0$.
8. Решить уравнение $D = 0$ относительно переменных c_{ij}.

В качестве примера рассмотрим, какие права доступа необходимо исключить из R_0 для предотвращения утечки права $(A, sw, write_r)$ в рассмотренном выше примере (рис. 4).

В результате применения алгоритма 14 и переименования переменных получим уравнение $c_1c_2c_3c_4 = 0$, где c_1 соответствует праву $(root, sw, write_r)$, c_2 – $(root, vuln\_ssh, write_r)$, c_3 – $(root, gw, read_r)$, c_4 – $(A, gw, write_r)$. Соответственно получаем следующие возможные решения:

1) $c_1 = 0$, $c_3 = 0$ – отключение *ssh*-службы;
2) $c_2 = 0$ – устранение уязвимости *vuln_ssh*;
3) $c_3 = 0$ – настройка правил фильтрации МЭ.

6. Формальная модель анализа защищенности сетей REM

В работе [8] представлено неформальное описание модели анализа защищенности сетей REM (Requirements, Effects, Modifies), используемой в топологическом сканере безопасности *NetSPA*. Основными положениями данной модели являются:

- описание атаки состоит из трех частей – предусловий атаки (requirements), постусловий атаки (effects), координат нарушителя (modifies);
- при описании используются переменные *host*, *software*, *user*, которые соответственно принимают значения (<*current_host*>, <*target_host*>), (*vendor*: *product name*: *common version*: *exact version*), (*none*, *nologin*, *user*, *root*);
- предусловиями атак служат предикаты: $host_i$ RUNS (ALSO_RUNS) $software_j$ [DOS] [TROJAN] – установка на узле $host_i$ программного обеспечения $software_j$, которое может быть установлено нарушителем или находиться в неработоспособном состоянии; $host_i$ CAN_CONNECT $host_j$ – связь между узлами; [$user_i$ on $host_i$] HAS_ACCESS $user_j$ on $host_j$ – пользователь $user_i$ с узла $host_i$ имеет доступ на узле $host_j$ с правами $user_j$;
- изменения состояний сети описываются предикатами $host_i$ RUNS $software_j$, [$user_i$ on $host_i$] HAS_ACCESS $user_j$ on $host_j$, TRUST_RELATIONSHIP – перехват данных в сети;
- изменения состояния нарушителя соответствуют изменениям <*current_host*> = <*target_host*>, <*current_user*> = <*target_user*>.

Опишем формальную REM ДП-модель анализа защищенности сетей на основе базовой и ФАС ДП-моделей.

Пусть $G = (S, E, R \cup A \cup F, H)$ – состояние системы $\Sigma(G^*, OP)$. Конкретизируем назначение некоторых элементов системы.

По определению являются: узел (компьютер) – сущностью-контейнером; коммуникационный порт – сущностью-контейнером или сущностью-объектом; файл – либо сущностью-контейнером, либо сущностью-объектом; пользователь или процесс пользователя – сущностью-субъектом.

Обозначим: E – множество сущностей системы, EC – множество узлов КС, CC – множество коммуникационных каналов, $CC \subset E$, $EC \subset E$; S – множество субъектов системы, при этом для каждого узла $c \in EC$ существует «пользователь ОС» $os_c \in S$ данного компьютера, U – множество пользователей, $U \subset S$;

R – множество ребер графа-состояния G, соответствующих правам доступа пользователей к сущностям;

A – множество ребер графа-состояния G, соответствующих доступам пользователей к сущностям;

F – множество ребер графа-состояния G, соответствующих информационным потокам между сущностями;

H: $E \to 2^E$ – функция иерархии сущностей, ее значения на множестве узлов EC соответствуют заданной в системе иерархии подчиненности компьютеров, и каждая сущность системы $e \in E$

- либо является узлом сети ($e \in EC$),
- либо размещена на некотором единственном для каждой сущности узле (для сущности e существует единственный компьютер $c \in EC$, такой, что $e < c$),
- либо является коммуникационным каналом ($c \in EC$),
- либо является пользователем ($e \in S$).

Аналогично [3] будем считать, что для каждого узла $c_1 \in EC$ определены доверенные пользователи $os_{c1} \in S$, обладающие правом доступа владения к каждой сущности, размещенной на данном компьютере.

В соответствии с положениями модели REM [8]:

- субъект-процесс p, выполняющийся от имени субъекта-пользователя u, будем считать функционально ассоциированным с ним, так как при наличии уязвимостей p нарушитель может получить права доступа пользователя u ко всем сущностям КС;
- наличие уязвимостей определяется по значению переменной *software* = (*vendor*: *product name*: *common version*: *exact version*) и дополнительно установленного программного обеспечения;
- уязвимые сетевые сервисы, позволяющие получать доступ к чувствительной информации (например, через нулевые сессии в ОС класса Windows), функционально ассоциированы с субъектом *nologin*;
- субъект *nologin* может быть функционально ассоциированным с субъектом-пользователем, информацию о котором он предоставляет.

Для состояния $G = (S, E, R \cup A \cup F, H)$ системы $\Sigma(G^*, OP)$ формально определим предикаты *run_soft*(x, y, G), *run_soft_dos*(x, y, G), *can_access_net*(x, y, z, v, G), *can_connect*(x, y, z, v,G), соответствующие предикатам RUNS (ALSO_RUNS), RUNS_DOS, HAS_ACCESS, CAN_CONNECT модели REM.

Предикат *run_soft*(x, y, G) является истинным тогда и только тогда, когда $y < x$ в состоянии G, где $y \in S$, $x \in EC$. При этом предполагается, что на узле x существует некоторый пользователь $u \in S$, что (u, y, own_r)$\in R$.

Предикат *can_connect*(x, y, z, v, G) является истинным тогда и только тогда, когда в состоянии G существует $c \in CC$, что (y, c, α_r)$\in R$, (v, c, α_r)$\in R$, где $\alpha_r \in \{read_r, write_r\}$, $y < x$, $v < z$, y, $v \in S$, x, $z \in EC$.

Предикат *run_soft_dos*(x, y, G) является истинным тогда и только тогда, когда предикат *run_soft*(x, y, G) истинен, а предикат *can_connect*(x, y, z, v, G) ложен для любых $v \in S$, $z \in EC$.

Предикат *has_access*(x, y, z, v, G) является истинным тогда и только тогда, когда в состоянии G истинен предикат *can_connect*(x, y, z, v, G) и (y, v, own_r)$\in R$, где $y < x$, $v < z$, y,$v \in S$, x, $z \in EC$.

Предикат *trust_relationship*(x, y, G) является истинным тогда и только тогда, когда для всех $c \in CC$, что (v, c, α_r)$\in R$, где $\alpha_r \in \{read_r, write_r\}$, $v < x$, истинно (y, c, $read_r$)$\in R$, где $y < x$, v, $y \in S$.

Изменение состояния нарушителя отображает последовательное получение доступа нарушителя к различным узлам сети с правами доступа различных пользователей. Если в состоянии G системы нарушитель, имея права доступа пользователя $a \in U$ на узле $b \in EC$, в результате выполнения последовательности преобразований система перейдет в состояние G', а нарушитель может получить права доступа пользователя $c \in U$ на узле $d \in EC$, то <*current_host*> = b, <*current_user*> = a, <*target_host*> = d, <*target_user*> = c. *Координатами нарушителя* в состоянии G' будем называть пару (<*target_host*>, <*target_user*>).

Определенные выше предикаты используются для проверки исходного состояния модели в соответствии с базой описания атак или становятся истинными в результате перехода из одного состояния в другое, что соответствует выполнению последовательности преобразований базовой или ФАС ДП-моделей.

Графом атак модели REM будем называть конечный помеченный ориентированный граф без петель $G_{\text{REM}} = (G^* \times EC \times U, E_{\text{REM}})$, в котором элементы множества $G^* \times EC \times U$ являются вершинами графа атак, а элемент $((G, x \times y), (G', z \times v)) \in E_{\text{REM}}$ тогда и только тогда, когда:

- в состоянии G истинны предикаты, соответствующие предусловиям атаки, в состоянии G' истинны предикаты, соответствующие постусловиям атаки;
- (z, v) – координаты нарушителя в соответствии с описанием атаки;
- существуют состояния $G_1, \ldots, G_N = (S_N, E_N, R_N \cup A_N \cup F_N, H_N)$ и правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G \vdash_{op1} G_1 \vdash_{op2} \ldots {}_{opN} G_N = G'$, где $N \geq 0$, и если правило op зависит от сущности $e \in E \backslash CC$, то $e < x$ или $e < z$.

Ниже показан граф атак модели REM сети, изображенной на рис. 2, где состояниям G, G_1, G' соответствуют графы доступов, изображенные на рис. 3, *а*, *в*, *г* соответственно.

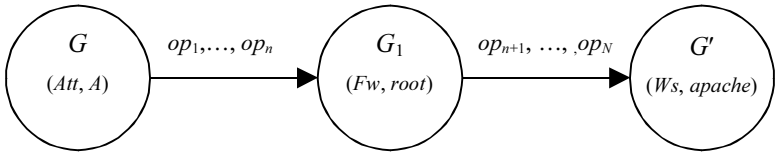


Рис. 5

7. Формальная модель анализа защищенности сетей VTG

Shahriary, Sadoddin, Jalili и Zakeri [2] на основе подходов, используемых в классической модели Take-Grant, предложили модель VTG (vulnerability Take-Grant) для анализа защищенности сетей. Модель VTG является расширением модели Take-Grant, в котором вводятся дополнительные права доступа, понятие уязвимости и новые правила преобразования – vulnerability rewriting rules.

Опишем основные положения модели VTG:

- каждому узлу сети соответствует некоторое множество известных уязвимостей;
- возможны следующие права доступа: h – размещения, x – выполнения, o – владения, r, w, t, g – как в классической модели Take-Grant;
- уязвимость сущностей обозначается с помочью их метки;
- рассматриваются 3 класса уязвимостей – переполнение буфера, слабые пароли, отношение доверия;
- для первых двух классов вводятся новые правила преобразования состояния модели: *buffer overflow* (*bof_rule*) и *password cracking* (*pc_rule*) rewriting rule;
- для проверки истинности предиката *can_access*(α, x, y, VTG_0) строится замыкание графа доступов;
- предикат *can_access*(α, x, y, VTG_0) является истинным тогда и только тогда, когда существуют состояния $\text{VTG}_1, \ldots, \text{VTG}_N$ и правила преобразования состояний $op_1, \ldots, op_N \in \{bof\_rule, pc\_rule, take\}$, такие, что $\text{VTG}_0 \vdash_{op1} \text{VTG}_1 \vdash_{op2} \ldots \vdash_{opN} \text{VTG}_N$ и ребро (x, y) помечено α в VTG_N, где $N \geq 0$.

Опишем формальную VTG ДП-модель анализа защищенности сетей на основе базовой и ФАС ДП-моделей. Неформально, под уязвимостью принято понимать некоторое нежелательное свойство КС, которое может быть использовано нарушителем при реализации атаки и может привести к осуществлению угрозы. *Уязвимостями* будем называть сущности, функционально ассоциированные с другими субъектами КС, если при реализации информационных потоков к ним происходит нарушение безопасности. Субъекты, имеющие такие сущности, назовем *уязвимыми*. По определению уязвимость является сущностью-объектом КС.

Будем использовать введенные ранее обозначения. Через V обозначим множество уязвимостей системы.

Правам r, w, x модели VTG соответствуют права доступа $read_r$, $write_r$, $execute_r$ ДП-моделей; вместо прав t, g, o используется право own_r, дуге в VTG-графе из вершины x в y, помеченной правом h, соответствует отношение $y<x$ в ДП-модели.

В соответствии с положениями модели VTG:

- при активации пользователем *u* некоторого процесса *p* последний наследует все права пользователя *u*;

- уязвимости процесса, связанные с переполнением буфера, функционально ассоциированы с пользователем, от имени которого данный процесс запущен;

- уязвимость вида «слабый пароль» функционально ассоциирована с пользователем-владельцем пароля, что позволяет проводить анализ условий передачи прав доступа и реализации информационных потоков по памяти, но не по времени.

Вместо правил *bof_rule* и *pc_rule* модели VTG будем использовать правило преобразования *control*() *ФАС* ДП-модели. Если пользователь *u* доверяет пользователю *v* в некотором состоянии $G = (S, E, R \cup A \cup F, H)$, то $(v, u, own_r) \in R$. Для построения замыкания VTG ДП-модели применим алгоритм 8 построения *control*-замыкания ФАС ДП-модели. *Граф атак* модели VTG – замыкание графа доступа VTG ДП-модели.

ЛИТЕРАТУРА

1. *Колегов Д.Н.* Проблемы синтеза и анализа графов атак // Вестник ТГУ. Приложение. 2007. № 23. С. 180 – 188.
2. *Shahriary H.*, *Sadoddin R.*, *Jalili R.*, *Zakeri R.* Network vulnerability analyses thorough vulnerability Take-Grant model // ce.sharif.edu/~shahriari/publications/ICICS2005.pdf.
3. *Девянин П.Н.* Анализ безопасности управления доступом и информационными потоками в компьютерных системах. М.: Радио и связь, 2006. 176 с.
4. *Bishop M.*, *Frank J.* Extending the Take-Grant protection system // http://citeseer.ist.psu.edu/frank96extending.html.
5. *Девянин П.Н.* Модели безопасности компьютерных систем: Учеб. пособие для студ. высш. учеб. заведений. М.: Издательский центр «Академия», 2005. 144 с.
6. *Jajodia S.*, *Noel S.*, *O'Berry B.* Managing Cyber Threats: Issues, Approaches and Challenges, ch. Topological Analysis of Network Attack Vulnerability. Kluwer Academic Publisher, 2003.
7. *Jajodia S.*, *Noel S.*, *et al.* Efficient Minimum-Cost Network Hardening Via Exploit Dependency Graphs // Proceedings of the 19th Annual Computer Security Applications Conference, Las Vegas, NV, USA, December 2003.
8. *Artz M.* NETspa, A Network Security Planning Architecture, M.S. Thesis. Cambridge: Massachusetts Institute of Technology, May 2002.

2008 Математические основы компьютерной безопасности № 1(1)

DOI 10.17223/20710410/1/13
УДК 004.56(06)

СОВРЕМЕННЫЕ МОДЕЛИ И МЕХАНИЗМЫ ЗАЩИТЫ ИНФОРМАЦИИ

М.М. Кучеров, И.Н. Кирко, А.А. Муллер

Сибирский федеральный университет, *г. Красноярск*

E-mail: kucherov@fivt.krgtu.ru

Разработана решеточная модель информационной безопасности, суть которой состоит в использовании общего уровня для целостности и доступности. Уровни, характеризующие целостность и доступность, совместно указывают на объективное качество в модели ценности информации.

***Ключевые слова*:** *мандатный контроль и управление доступом*, *закон Гроша*, *эффективность доступа.*

Сегодня многие организации объединяют ресурсы. Такое объединение выгодно всем участникам информационного обмена. При объединении фирм также происходит объединение общих электронных документов, файлов, баз данных и т.д. Для того чтобы обеспечить безопасность информации, необходим процесс управления информацией и разграничение доступа внутри и за пределами каждой организации.

Управление информацией предполагает, что известен объем информации, требуемый для управления производством. Невозможно, фактически, подробно перечислить информационные объекты, но можно назвать основные элементы информации, необходимые для производственных операций. Элемент информации представляет собой основную часть информации. Он может появляться в разных местах в ментальной, письменной или электронной форме.

Основной элемент информации имеет следующие свойства:

- является существенным для производства;
- создается первичной производственной функцией;
- может находиться в разнообразных сочетаниях с другой информацией.

Список основных элементов информации, существенных для производства, указывает на элементы информации, которые являются объектами управления и контроля. Решение о том, будет или не будет предприятие контролировать соответствующий элемент информации, зависит от его значения. В случае создания новых элементов информации или использования уже существующих элементов руководство предприятием должно предпринять необходимые действия для того, чтобы эти элементы информации имели соответствующее значение, что позволило бы при необходимости контролировать их надлежащим образом.

Запросы на информацию постоянно возрастают с увеличением объемов производства. Закон Гроша утверждает, что увеличение в n раз отдачи от инвестиций в компьютерное оборудование возможно при увеличении в n^2 раз скорости обработки информации, т.е. сокращении в соответствующее число раз времени доступа к информации [1, 8]. Почти невозможно предвидеть требования на доступ к центральным базам данных, а также приложения, с помощью которых будет обработана полученная информация. Следовательно, важно, чтобы элементы информации были отделены от производственных приложений и находились в форме, позволяющей легко их использовать для разных приложений и целей.

1. Определения

Вначале дадим необходимые определения [2 – 6]. Мандатное управление доступом есть разграничение доступа субъектов к объектам, основанное на характеризуемой меткой конфиденциальности информации, содержащейся в объектах и официальном разрешении (допуске) субъектов обращаться к информации такого уровня конфиденциальности.

Под уровнем конфиденциальности будем понимать иерархический атрибут, который может быть ассоциирован с сущностью компьютерной системы для обозначения степени ее критичности в смысле безопасности. Этот атрибут может обозначать, например, степень ущерба от нарушения безопасности в компьютерной системе или чувствительность (чувствительность — характеристика ресурса, которая определяет его ценность или важность и может учитывать его уязвимость).

Степень доверия — это атрибут, определяющий уровень конфиденциальности субъекта. Чем выше степень доверия субъекта, тем к более секретной информации он имеет доступ. Чем выше конфиденциальность объекта, тем более секретная информация хранится в нем.

Определим функции, отображающие субъектов и объекты системы, на уровни конфиденциальности — допуск (*clearance*) и гриф (*classification*). Областью значений каждой из этих функций является решетка уровней конфиденциальности L с линейным отношением частичного порядка. Функция *clearance* определе-

на на множестве субъектов *S*, а *classification* — на множестве объектов *O*. Эти функции записываются следующим образом:

$$clearance\ (s) = l_s; \tag{1}$$

$$classification\ (o) = l_o. \tag{2}$$

Кроме уровней конфиденциальности, можно ввести множество категорий. Каждая категория описывает соответствующий тип информации. Примерами категорий могут служить {«Для прессы», «Основной», «Стратегический» и т. п.}. Объект, которому присвоено несколько категорий, содержит информацию, соответствующую этим категориям.

Множество уровней конфиденциальности образуют решетку по отношению к операции (<). Решетка (SC, $\leq$) определяется следующим образом:

SC – конечное множество уровней конфиденциальности;

$\leq$ – бинарное отношение частичного порядка для уровней конфиденциальности SC.

Отношение ($\leq$) рефлексивно ($A \leq A$), транзитивно ($A \leq B$ & $A \leq C \Rightarrow A \leq C$) и симметрично ($A \leq B$ & $B \leq A \Rightarrow A = B$).

При этом существует наибольшая верхняя граница в SC, т.е. для каждого A и B в SC существует класс $C = \max(A, B)$, такой, что:

1) $A \leq C$ и $B \leq C$;

2) $A \leq D$ и $B \leq D \Rightarrow C \leq D$ для любого D из SC.

Для каждого непустого подмножества $S = \{A_1, A_2, ..., A_n\}$ из SC существует единственный элемент $S = \max(A_1,..., A_n)$.

Можно определить также наименьшую нижнюю границу в SC. Для каждого A и B в SC существует единственный класс $E = \min(A, B)$, такой, что:

1) $E \leq A$ и $E \leq B$;

2) $D \leq A$ и $D \leq B \Rightarrow E \leq D$ для любого D из SC.

Для каждого непустого подмножества $S = \{A_1, A_2, ..., A_n\}$ из SC существует единственный элемент $S = \min(A_1,..., A_n)$.

2. Решеточная модель ценности

Решения относительно качества информации следует принимать на основе формальной модели, которая основана на следующих двух критериях: субъективной и объективной ценности.

Субъективная ценность соотносится с размером нанесенного ущерба в случае, если информация окажется известной лицам, неавторизованным для ее получения. Здесь ущерб возникает из качеств, присущих самой информации. Объективная ценность соотносится с размером нанесенного ущерба в случае, если информация окажется недоступной или имеет ненадлежащую целостность. Это – объективный критерий, зависящий от событий или спецификаций вне самой информации. Например, если финансовые документы, которые по закону должны храниться в течение ряда лет, окажутся утраченными, предприятие может быть подвергнуто штрафу.

Отдельные элементы информации согласно этой модели попадают в обе категории. Например, ведомости по персоналу, в которых указаны суммы компенсации сотрудникам, имеют субъективную и объективную ценность.

В табл. 1 приведен пример классификационной политики.

Таблица 1

Пример классификационной политики в области информации

Субъективные классификации	Метка конфиденциальности		
	Для служебного пользования	Конфиденциальная	Персональная
Определение	Раскрытие может нанести в перспективе ущерб экономике предприятия	Раскрытие может нанести серьезный ущерб экономике предприятия	Раскрытие может негативно повлиять на сотрудников или претендентов на должность
Объективные классификации	На хранении	На текущем контроле	
Определение	Ненадлежащее качество может привести в перспективе к серьезным правовым или экономическим последствиям	Ненадлежащее качество может нанести серьезный ущерб экономике предприятия	

В реальном мире существуют группы предприятий, интересы которых взаимосвязаны. Поместим их в классы конфликтных интересов. В каждом из классов находится определенное количество компаний, конкурирующих на рынке. В каждой компании, о которых было сказано выше, работают консультанты, которые имеют доступ к информации своей компании. Возможны ситуации, когда консультант из одной компа-

нии имеет доступ к сведениям других компаний. В каждой компании имеется информация, которая классифицируется на основе модели ценности. Информация содержится в объектах модели безопасности.

Опишем структуру модели безопасности и определим ее элементы, компоненты, свойства и правила. На рис. 1 изображена структура информационных ресурсов компаний.

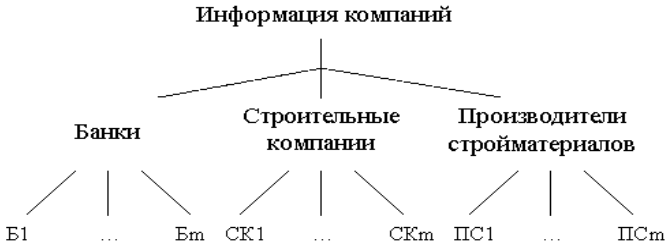


Рис. 1. Субъектная классификация информации предприятий

В каждой компании присутствует общедоступная и конфиденциальная информация. Информация делится на информацию с высоким уровнем целостности и на информацию с низким уровнем целостности (т.е. информацию, которую можно изменять). Разбиение всей информации на два уровня целостности необходимо для защиты документов от модификации, что соответствует структуре модели Биба [2].

Таким образом, объекты в данной модели описываются следующими свойствами:

- номер (или номера) компании, к которой эта информация принадлежит;
- уровень конфиденциальности (общедоступный или конфиденциальный);
- уровень целостности (низкий или высокий).

Для субъектов имеются следующие свойства:

- номера компаний, информация которых доступна субъекту;
- уровень конфиденциальности доступной информации.

Каждому субъекту и объекту в системе соответствует метка, которая отражает его свойства. Определим следующую структуру для политики безопасности:

1. Имеется несколько классов конфликтных интересов, например банки, строительные компании, производители стройматериалов.

2. Каждый класс содержит m_k компаний, как изображено на рис.1. Для каждого объекта в системе определена метка безопасности, которая представляет n-элементный вектор $\{i_1, i_2, \ldots, i_n\}$, где каждый элемент $i_k \in SC$ или $i_k = \perp$ (читается, как самый нижний) для $1 \le k \le n-2$ (последние два места отведены для атрибутов целостности и доступности). Объект, обозначенный $\{i_1, i_2, \ldots, i_n\}$, интерпретируется как содержащий информацию от компании i_1 (одного из m_1 банков), информацию от компании i_2 (одной из m_2 строительных компаний) и т.д. Если элемент вектора представляет $\perp$, то это означает, что объект не содержит критической информации ни из одной компании в соответствующем классе конфликтных интересов.

3. Пусть $S = \{s_1, s_2, ..., s_{ns}\}$, где S – множество субъектов, s_i – отдельные субъекты системы, ns – количество субъектов в системе.

Приведем пример субъекта s_1 для структуры, представленной на рис. 1: $s_1 = (1, 1, 1)$ – метка субъекта s_1, в ней содержится информация о том, что данный субъект имеет доступ к конфиденциальной информации банка 1 (Б1), строительной компании 1 (СК1), а также производителей стройматериалов 1 (ПС1).

4. Пусть $O = \{o_1, o_2, ..., o_{no}\}$, где O – множество объектов, o_i – объекты системы, no – количество объектов в системе.

Рассмотрим объект o_1 на рис. 1. Метка объекта $o_1 = (1, \perp, \perp, вц, вд)$ говорит о том, что информация этого объекта (документа) содержит конфиденциальные данные банка 1 (Б1) и не содержит конфиденциальной информации по другим классам конфликтных интересов, т.е. строительных компаний и производителей строительных материалов. Объекту присвоены метки высокой целостности и доступности.

Метки субъектов и объектов необходимы для того, чтобы можно было определить те объекты, к которым данные субъекты имеют доступ, и действия, которые они могут совершать. Все это определяют правила чтения и записи.

Правило чтения:

Субъект S может читать объект O, только если:

- метка S доминирует над меткой O.

Правило записи:

Субъект S может записывать в объект O, только если:

- метка O доминирует над меткой S.

Например, субъект s $\{2, 1, \perp\}$ может прочитать объект o $\{\perp, 1, \perp, вц, вд\}$, так как метка субъекта s доминирует над меткой объекта o и выполняется условие, разрешающее чтение. А субъект s $\{3, \perp, \perp\}$ не может прочитать объект o $\{4, 2, 1, нц, нд\}$, потому что они принадлежат разным классам конфликтных интересов.

Информация от субъекта s {2, 1, ⊥} может быть направлена в объект o {2, 1, 1, *нц*, *нд*}, потому что $o \geq s$, выполняется условие правила записи, а из субъекта s {1, 2, 3} в объект o {2, 3, 1, *нц*, *нд*} – не может, так как эти субъект и объект несопоставимы. Они не находятся в одном классе, информация принадлежит одной из конкурирующих компаний, а пользователь – другой.

Дополнительно в пределах одной компании в метке субъекта может быть добавлена роль субъекта в системе. Такая метка с записью о роли может быть использована только для потоков информации внутри одной компании, в которой назначена эта роль. Роли присваивается уникальный номер, и он указывается в метке соответствующего субъекта. Например, субъект s, соответствующий генеральному директору компании, может выглядеть следующим образом: s (1, 3, 2; 1).

Дополним это описание с учетом доступности. Как уже говорилось, закон Гроша утверждает, что ключ к увеличению отдачи от инвестиций заключается в сокращении времени доступа к информации. В таком случае необходимо размещение субъектов и объектов с высокой доступностью на высшей ступени иерархии целостности, т.е. «на текущем контроле» в модели ценности (табл. 1). В результате субъекты и объекты с высокой доступностью являются также и высокоцелостными, а компоненты с низкой доступностью не всегда являются таковыми, т.е. могут быть модифицированы. Таким образом, правила NWU (нет записи вверх) и NRD (нет чтения вниз) в структуре модели Биба соответствуют решеточной модели безопасности Белла и Лападула, так как чтение снизу высокоцелостных файлов в иерархии модели целостности Биба происходит быстро. Аналогично быстро происходит и запись информации вниз, т.е. «на хранение». В свою очередь, противоположно направленные информационные потоки, такие, как перевод информации из категории «на хранении» в категорию «на текущем контроле», и доступ высокоцелостных субъектов, в т.ч. компьютерных процессов, к программам и данным ненадлежащего качества подлежат контролю.

На рис. 2 разрешены два информационных потока: поток по записи (*write*) снизу вверх и поток по чтению сверху (*read*) – это работа с файлами операционной системы, официальными утвержденными отчетами, справочными данными и тому подобной информацией с высоким уровнем целостности. Перезапись официальной информации должна быть ограничена и строго контролироваться (NWD), поскольку она препятствует высокопроизводительной работе.

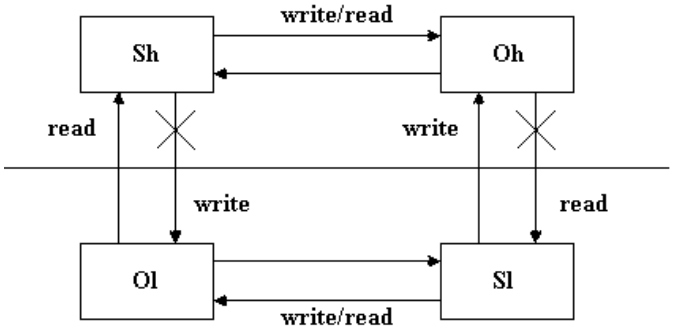


Рис. 2. Дополнение модели Белла и Лападула с учетом доступности. Высокий уровень конфиденциальности объединен с низким уровнем доступности, а низкий уровень конфиденциальности – с высоким уровнем доступности

Точно так же должен быть ограничен поток по чтению снизу (NRU) в модели Белла и Лападулы, поскольку он не только приводит к несанкционированному доступу к информации, но и замедляет высокопроизводительную обработку данных.

Процедуры коррекции информации с высоким уровнем доступа должны инициироваться записью вверх, на более конфиденциальный уровень. Таким образом, достигаются две цели: информация становится недоступной для рядовой обработки, и на более секретном уровне к ней получают доступ субъекты, которые могут ее редактировать.

Перевод информации с низкого на более высокий уровень доступности означает деклассификацию информации, т.е. ее санкционированное распространение, и должен проводиться на основе, во-первых, установления правильного форматирования и представления вновь публикуемой информации, во-вторых, проверки ее своевременности и соответствия реалиям деловых процессов и, в-третьих, под ответственность принципала, от лица которого выполняется эта операция.

Необходимо отметить, что процесс редактирования информации должен осуществляться по завершении любой иной ее обработки, в противном случае в системе одновременно окажется два «официальных» источника данных, что является нарушением.

На практике это позволяет разместить важные системные файлы в верхней части иерархии модели Биба. Это защищает доступность эталонных файлов и проверенных данных от обычных пользователей, поскольку

правило NWU не позволяет им осуществить запись в важные документы и, тем самым, исправлять официальные данные. Кроме того, если рассматривать исполнение как чтение, то высокопроизводительные процессы компьютерной системы, такие, к которым и относится закон Гроша, не могут оперировать документами и данными вне высшего круга целостности. Это обеспечивает дополнительную защиту целостности в компьютерной системе.

Данная схема обеспечивает защиту системных файлов от троянских программ. Если троянская программа находится на одном из нижних уровней в иерархии модели Биба, то она не сможет исказить системные файлы за счет правила NRD. Таким образом, осуществляется защита производительности и целостности от троянских программ. Очевидно, что такое объединение моделей может также осуществлять защиту конфиденциальности для верхних уровней определенной иерархии и защиту эффективного доступа для нижних уровней в модели Белла и Лападула.

Данная модель может объединить в себе как модель решеточного доступа (когда пользователям назначен доступ к конфиденциальным документам), так и модель дискреционного доступа (когда в матрице доступа в рамках одного предприятия может быть составлено соответствие должностей и документов, к которым пользователь, владеющий данной меткой, может иметь ограниченный доступ). Ее можно использовать при обмене информацией с другими компаниями с теми же метками, за исключением последней записи о роли субъекта, которая оставляется только для внутреннего документооборота, когда каждой роли будет соответствовать набор прав по доступу к информации. Соответствие ролей и уникальных номеров можно сохранять в таблице или матрице (табл. 2). Пример матрицы доступа приведен в табл. 3.

Таблица 2

Соответствие ролей их уникальным номерам в системе

Роль	Код
Генеральный директор	1
Заместитель директора	2
Главный бухгалтер	3

Таблица 3

Пример матрицы доступа

Роль	Права доступа	
	R	*W*
1	1	1
2	1	0
3	1	1

В матрице доступа в табл. 3 описаны права на запись *W* и на чтение *R*. Разновидность прав может быть определена, исходя из задач, которые ставятся перед пользователями. Из табл. 2 и 3, видно, например, что генеральный директор может читать и записывать в данный файл, а заместитель директора данный файл может только читать.

Матрица доступа в табл. 3 соответствует какому-то одному документу или файлу. Такая матрица должна быть создана для каждого файла, если используется дискреционная модель разграничения доступа.

3. Реализация модели

Переходим к описанию предлагаемого решения по реализации такой модели безопасности на практике. В качестве места для хранения информации об объектах и субъектах системы была выбрана база данных. Тем самым обеспечиваются все необходимые требования по обращению с информацией и пользователями, поскольку существуют готовые решения по защите баз данных и способах разграничения доступа к их ресурсам.

Опишем модель данных такой базы данных и поля таблиц, которые она будет содержать. База данных делится на две части: первая часть содержит информацию об объектах, другая часть содержит информацию о субъектах модели безопасности.

Информация каждой отдельной компании находится в отдельной таблице или таблицах, исходя из содержания такой информации. Если имеется *m* компаний в системе, тогда в общем случае имеется *m* таблиц, содержащих информацию. Приведем пример, в котором опишем структуру такой таблицы и дадим определение полей, которые должны присутствовать в обязательном порядке.

1. Идентификатор объекта – это уникальное поле, первичный ключ таблицы, необходим для того, чтобы обеспечить однозначную идентификацию записей таблицы, предотвратить повторение значений ключа, ускорить выполнение запросов к базе данных, установить связи между отдельными таблицами базы данных.

2. Принадлежность информации к другим компаниям – поле, которое содержит информацию о том, к информации какой компании имеет отношение данный объект. Уже говорилось о том, что информация в такой системе может быть доступна сотрудникам из других компаний. По значению этого поля можно судить об уровне конфиденциальности данного объекта.

3. Уровень целостности (объективный признак) – это поле, по которому можно судить о том, возможна или нет модификация объекта.

Уровень доступности – определяется уровнем целостности согласно приведенной модели безопасности.

Информация – данные, которые содержит объект.

Перечисленные выше поля таблицы являются обязательными, они обеспечивают связь объектов с субъектами. В табл. 4 содержится информация об объекте компании 1 класса конфликтных интересов 1.

Описания субъектов компании также хранятся в отдельной таблице. Таблица содержит информацию о метках безопасности субъектов, которые позволяют увидеть, к какой информации и у какого субъекта имеется доступ. Опишем значение полей, содержащихся в таблицах субъектов.

Таблица 4

Экземпляр таблицы объектов

Объекты: 1, 1
Идентификатор: 757
Принадлежность: 1,3,2
Целостность: 0
Информация: Текст
Дополнительное

1. Идентификатор субъекта – это уникальное поле, первичный ключ таблицы, он необходим для того, чтобы обеспечить однозначную идентификацию записей таблицы, предотвратить повторение значений ключа, ускорить выполнение запросов к базе данных, установить связи между отдельными таблицами базы данных.

2. Логин – уникальное имя пользователя, которое необходимо для прохождения процесса аутентификации в системе.

3. Пароль – поле, содержащее хешированное значение пароля пользователя.

4. Права доступа – поле, содержащее метку безопасности.

5. Информация – информация о пользователе, например его имя.

В табл. 5 содержится информация о субъекте компании 2 класса конфликтных интересов 2.

Таблица 5

Экземпляр таблицы субъектов

Субъекты: 2,2
Идентификатор: 867
Логин: Alex
Пароль: 1G4f~2
Метка: 1, 2, 3
Личное: Алексей

В заключение были рассмотрены существующие модели безопасности и на основе модели Брюера и Нэша [7] была построена объединенная модель безопасности, которая включила модели Белла и Лападула, Биба, контроля доступа, базирующегося на ролях, а также модели дискреционного доступа. Предложена модель ценности информации, в которой имеются только две характеристики производственной информации: субъективная, которая указывает на конфиденциальность, и объективная, которая связывает вместе целостность и доступность.

При работе над составлением требований к системе защиты использовался стандарт Банка России [3]. Это обусловлено проблемами, связанными с утечкой информации по внутренним каналам организаций, а также необходимостью защиты передаваемой информации между различными компаниями. Вместе с тем стандарт Банка России является на сегодняшний день главным документом в сфере информационной безопасности для работы организаций банковской сферы и кредитных организаций, которые и являются субъектами описываемой модели. Кроме того, стандарт рассматривает комплексно вопрос защиты и хотя и носит рекомендательный характер, но уже в ближайшем будущем, возможно, будет являться обязательным для применения. Поэтому предложенное решение включило требования по защите информации, предъявляемые стандартом безопасности Банка России.

ЛИТЕРАТУРА

1. *Олифер В.Г.*, *Олифер Н.А.* Компьютерные сети. Принципы, технологии, протоколы. Питер, 2007.
2. *Корт С.С.* Теоретические основы защиты информации: Учеб. пособие. М.: Гелиос АРВ, 2004.
3. *Стандарт* Банка России СТО БР ИББС-1.0-2006 Обеспечение информационной безопасности организаций банковской системы Российской Федерации.
4. *Зегжда П.Д.* Теория и практика обеспечения информационной безопасности. М.: Яхтсмен, 1996.
5. *LaPadula L.*, *Bell D.* Secure Computer System: Mathematical Foundation, ESD-TR-73-278. V.1, MITRE Corporation.
6. *McLean J.* Secure models, Encyclopedia of software engineering. 1994.
7. *Brewer D.F.C.*, *Nash M.J.* The Chinese Wall Security Policy: IEEE Symposium on Security and Privacy. 1989. P. 215 – 228.
8. *Gardner W.D.* Author of Grosch's Law Going Strong At 87: TechWeb Technology News: techweb.com/wire/networking/160701379/

2008 Математические основы компьютерной безопасности № 1(1)
DOI 10.17223/20710410/1/14
УДК 681.322

РЕАЛИЗАЦИЯ ПОЛИТИК БЕЗОПАСНОСТИ В КОМПЬЮТЕРНЫХ СИСТЕМАХ С ПОМОЩЬЮ АСПЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ

Д.А. Стефанцов

Томский государственный университет

E-mail: d.a.stephantsov@gmail.com

Рассматривается аспектно-ориентированное программирование как средство реализации политик безопасности в компьютерных системах. Даётся краткий обзор аспектно-ориентированного языка программирования AspectJ с примером реализации аспектов безопасности с помощью этого языка. Вводятся понятия из областей объектно-ориентированного программирования и метаобъектных протоколов. Показывается, как на основе метаобъектного протокола может быть построено аспектно-ориентированное программирование. Даётся краткий обзор аспектно-ориентированного языка AspectTalk, разработанного автором статьи, а также рассматривается пример реализации простой политики безопасности с помощью этого языка.

Ключевые слова: *политика безопасности*, *аспектно-ориентированное программирование*, *объектно-ориентированное программирование*, *метаобъектный протокол*, *AspectJ*, *AspectTalk*, *MetaclassTalk*, *Smalltalk.*

В связи с проникновением компьютерной техники в производство, образование и повседневную жизнь резко возрастает потребность в защите вычислений от случайных или умышленных вмешательств. В данной статье будем рассматривать только умышленные вмешательства. Поэтому *угрозу* можно определить как цель злоумышленника, а *атаку* – как способ достижения этой цели.

Безопасностью называется состояние защищённости системы от некоторых видов угроз и/или атак. Это определение подразумевает, что безопасность – относительное понятие, т.е. та или иная защита – это защита от конкретных атак (и, возможно, от некоторых угроз) [1].

Как правило, для защиты компьютерных систем поступают следующим образом: вводятся формальные определения компьютерной системы и некоторой угрозы. Затем предъявляется ряд требований, соблюдение которых влечёт невозможность реализации данной угрозы в данной системе. Такие требования формулируются в виде *теорем безопасности* [2]. Наличие формальных требований к системе в большинстве случаев даёт возможность применения в реальных компьютерных системах алгоритмов, приводящих к соблюдению этих требований.

Политикой безопасности будем называть ряд требований, предъявляемых к компьютерной системе с тем, чтобы привести её в состояние безопасности. *Реализацией политики безопасности* будем называть реализацию на каком-либо языке программирования алгоритмов, обуславливающих соблюдение требований политики безопасности.

Реализации политик безопасности можно встретить во многих компьютерных системах – от операционных систем общего назначения [3] до узкоспециализированных Web-систем хранения данных. Несмотря на годы практики, большинство методов реализации политик безопасности являются грубым нарушением принципа «разделения аспектов» изучаемого предмета [4].

Некоторые исследователи способов проектирования программного обеспечения (ПО) сходятся во мнении, что данное нарушение – следствие ограничений, накладываемых современными средствами реализации, в частности существующими парадигмами программирования [5, 6]. В этих же источниках указывается один из возможных путей решения данной проблемы – аспектно-ориентированное программирование (АОП). Данная статья посвящена введению в это средство разработки ПО и реализации с его помощью политик безопасности компьютерных систем.

Процесс реализации защищённого приложения традиционными методами предполагает как совместное проектирование политик безопасности и основной части приложения, так и их совместную реализацию. Если впоследствии появится необходимость в изменении политики безопасности, то внедрение этих изменений будет осложнено необходимостью анализа, повторного проектирования и реализации всех точек пересечения (модулей) аспекта безопасности и главной части приложения. Отметим, что изменение политики безопасности зачастую является единственной адекватной мерой противодействия новым видам атак и/или угроз, и, несмотря на это, политики безопасности современных компьютерных систем остаются неизменными. Вероятно, причиной тому служит излишняя сложность внесения изменений.

АОП является расширением существующих парадигм программирования: аспектно-ориентированный язык обладает как конструкциями, характерными для традиционных языков, так и конструкциями, с помо-

щью которых описываются аспекты приложения. При использовании АОП основная часть приложения и его аспекты проектируются и реализуются независимо друг от друга и объединяются в одно целое в автоматическом режиме. Разработка основной части осуществляется, как правило, объектно-ориентированными конструкциями языка программирования, а аспектов – дополнительными, аспектно-ориентированными, конструкциями языка. Объединение этих частей в приложение производится либо виртуальной машиной, либо механизмами текстовых подстановок. В случае необходимости внесения изменений в политику безопасности они отразятся только на исходном коде аспектов приложения, но не на основной его части. Поскольку аспекты в АОП реализуются изолированно, эти действия производятся относительно простым для разработчика образом.

1. Аспектно-ориентированное программирование

По мнению авторов АОП, основным принципом разработки новых парадигм программирования служил принцип эффективной декомпозиции приложения на модули [7]. Однако ни одна из существующих парадигм не предложила способа разделения реализации различных аспектов одного приложения.

Понятие «аспект» понимается так же, как в [4], а именно как точка зрения на изучаемый предмет, рассматриваемая в изоляции от других точек зрения. Так, один из аспектов приложения (помимо основного назначения) – это безопасность (защищённость от конкретных атак).

Если, в силу декомпозиции системы, один и тот же модуль должен реализовывать алгоритмы двух или более аспектов, то говорят о пересекающихся аспектах. АОП – это парадигма программирования, целью которой является введение уровня модульности, позволяющего реализовывать различные аспекты системы изолированно друг от друга, независимо от того, пересекаются ли они или нет при первоначальной декомпозиции [5].

Реализация ПО методом АОП заключается в раздельном описании каждого аспекта системы (в том числе политики безопасности) и указании способа соединения различных аспектов в единое приложение [7]. Соединение различных аспектов в приложение осуществляется автоматически, средствами аспектно-ориентированных средств разработки.

АОП не является заменой какой-либо другой парадигмы программирования, но её расширением [5]. Типичный процесс создания приложения с помощью аспектно-ориентированных технологий выглядит следующим образом: производится декомпозиция системы на модули привычными (не аспектно-ориентированными) методами, выделяются модули, по которым пересекаются несколько аспектов приложения, один аспект объявляется главным (он выполняет основное назначение системы; такой аспект в будущем будем называть главным) – он реализуется привычными средствами (без АОП), прочие аспекты реализуются с помощью АОП.

Авторами АОП был взят за основу объектно-ориентированный язык программирования Java. Соответствующая аспектно-ориентированная версия языка носит название AspectJ [8].

Для иллюстрации применения АОП рассмотрим следующий фрагмент программы на языке Java (листинг 1).

```
public class Process {
    ...
    public void readAndPrint(File f) {
        if (this.id == f.id)
            System.out.println(f.getInfo());
        else
            System.out.println("This is not my file!");
    }
    ...
}
```

Листинг 1. Фрагмент программы на языке Java

Как видно, этот модуль (метод readAndPrint объекта класса Process) реализует алгоритмы сразу двух аспектов – главного аспекта (вывод на экран содержимого объекта класса File) и действий, связанных с обеспечением безопасности (проверка совпадения идентификаторов).

Требования, предъявляемые двумя аспектами к одному модулю, могут быть изолированы друг от друга средствами аспектно-ориентированного языка AspectJ, являющегося расширением языка Java (см. листинг 2).

В данном случае реализация чтения из файла изолирована от проверки совпадения идентификаторов. Служебное слово aspect объявляет новый аспект Security. Конструкция around указывает, что вместо метода void Process.readAndPrint (File) должен быть выполнен алгоритм, указанный далее в фигурных скобках. Алгоритм состоит в проверке совпадения идентификаторов. В случае успеха с помощью служебного слова proceed производится исполнение метода readAndPrint объекта p класса Process с аргументом f класса File.

```
public class Process {
    ...
    public void readAndPrint(File f) {
        System.out.println(f.getInfo());
    }
    ...
}
...
public aspect Security {
    ...
    void around(Process p, File f):
        call(void Process.readAndPrint(File)) && args(f) && target(p) {
            if (f.id == p.id)
                proceed(p, f);
            else
                System.out.println("That's not my file!");
    }
    ...
}
```

Листинг 2. Фрагмент программы на языке AspectJ

Следует отметить, что АОП не обязательно реализуется лингвистически, то есть в виде языка программирования. В [9] описываются другие подходы, в частности создание АОП-подобной архитектуры приложения. О различных подходах к реализации АОП можно прочитать также в [10–12].

2. Реализация аспектно-ориентированных средств разработки

В настоящее время возраст аспектно-ориентированных технологий превышает десять лет [13]. За это время образовалось сообщество исследователей, изучающих различные вопросы аспектно-ориентированного программирования [14]. Исследования поддерживаются крупными организациями, такими, как ACM.

Опишем один из подходов, применявшийся в [15, 16]. Первоначально аспектно-ориентированное программирование являлось расширением объектно-ориентированного программирования (ООП). Дадим определение *объекта* в объектно-ориентированной вычислительной системе.

Пусть K – непустое множество, которое будем называть множеством селекторов. Объект O есть тройка (S, M, s), где S – непустое множество, называемое множеством состояний объекта, s O S – начальное состояние объекта, M – множество пар (k, m), где k O K, m – алгоритм, представляющий собой последовательность команд для некоторой машины (которую будем называть *виртуальной машиной*) одного из следующих четырёх типов. Будем обозначать $s(t)$ O S состояние объекта O в момент времени t.

1. Команда изменения состояния. В результате выполнения команды состояние объекта изменится с $s(t)$ на $s(t + 1)$.

2. Команда посылки сообщения. Пусть A, B – объекты, $A = (S_A, M_A, s_A)$, $B = (S_B, M_B, s_B)$. Пусть (k_A, m_A) O M_A, (k_B, m_B) O M_B и в алгоритме m_A встретилась команда посылки сообщения k_B объекту B. В результате выполнения этой команды виртуальная машина передаст управление первой команде алгоритма m_B.

3. Команда возврата результата. Пусть $A = (S_A, M_A, s_A)$, $B = (S_B, M_B, s_B)$, (k_A, m_A) O M_A, (k_B, m_B) O M_B. Пусть объект A посылал сообщение k_B объекту B и в алгоритме m_B встретилась команда возврата результата. После выполнения этой команды состояние объекта A изменится с $s_A(t)$ на $s_A(t + 1)$, причём это изменение зависит от $s_A(t)$ и $s_B(t)$.

4. Команда порождения объекта. Пусть $C = (S_C, M_C, s_C)$ и (k_C, m_C) O M_C. Пусть команда порождения объекта встретилась в алгоритме m_C. В результате выполнения этой команды состояние $s(t)$ изменится на $s(t + 1)$, а также конструируется новый объект $O = (S, M, s)$, где S, M, s зависят от $s_C(t)$ – состояния объекта C в текущий момент времени.

Детали в этом определении уточняются в конкретной объектно-ориентированной модели вычислений (языке программирования). Более детальные определения объектно-ориентированного метода разработки программного обеспечения можно прочитать в [17], подробнее об ООП – в [18].

Одной из основных особенностей реальных объектно-ориентированных систем является то, что множества S и M не задаются для каждого объекта в отдельности, а указываются в специальных объектах-классах. Существует механизм наследования, иерархически упорядочивающий особые объекты (классы) в иерархическую структуру (в более сложном случае – в ациклический ориентированный граф). При этом, если объект O был порождён объектом-классом C_n и в иерархии наследования $C_n \to C_{n-1} \to \ldots \to C_1$ – путь от C_n к корню, то множество M объекта O есть объединение множеств методов, указанных в классах $C_1, \ldots, C_{n-1}, C_n$.

Подобные правила определения соответствия алгоритма селектору называются *метаобъектным протоколом* [19]. Метаобъектный протокол является одним из средств реализации АОП. Основой такой реализа-

ции является возможность поставить программный обработчик отправки определённого сообщения от определённого объекта к другому определённому объекту.

Если два аспекта пересекаются по одному программному модулю (методу некоторого объекта), то главный аспект реализуется объектно-ориентированным методом, а прочие аспекты реализуются методом перехвата посылки сообщения и запуска соответствующего алгоритма. Таким образом достигается раздельная реализация различных аспектов.

Как правило, для реализации такого перехвата вводится понятие метакласса. С одной стороны, метакласс – это класс класса как объекта (т.е. именно он ответственен за реализацию статических член-данных и член-функций в терминах языка C++), а с другой стороны, он управляет диспетчеризацией сообщений, пересылаемых объектам, являющимся экземплярами объекта-класса, который, в свою очередь, является экземпляром метакласса.

В случае, если транслятор с языка программирования переводит программу на некоторый промежуточный язык, выполняемый виртуальной машиной, подобный перехват сообщений реализуется средствами виртуальной машины. Таким образом, АОП реализовано в [15, 16], а также в языке AspectTalk. Однако существуют методы реализации АОП для компилируемых языков программирования.

3. Язык AspectTalk и пример реализации политики безопасности на этом языке

За основу языка AspectTalk взят язык Smalltalk [20, 21], к которому добавлен механизм метаклассов, аналогичный [15], с некоторыми дополнительными возможностями. А именно, множество объектов, к которым может быть произведено обращение из алгоритмов-перехватчиков, включает как объект, которому было послано сообщение, так и объект, пославший сообщение. Таким образом можно производить двустороннюю аутентификацию, что является необходимым условием реализации политик безопасности.

На листинге 3 показана программа, моделирующая поведение пользователей в системе. В строке 1 добавляются новые переменные alice и bob, в строках 2 и 3 они инициализируются объектами типа Process. В строках 4 и 6 эти объекты-процессы создают одноимённые файлы, а в строках 5 и 7 помещают в них свои конфиденциальные данные. Далее, в строках 8 и 9 alice пытается прочитать свой файл, а в строках 10 и 11 то же самое пытается сделать bob. Наконец, в строках 12 и 13 alice пытается прочитать файл bob'a.

```
 1 self addVariables: #(#alice #bob).
 2 alice <- Process new.
 3 bob <- Process new.
 4 alice execCreate: 'alice.txt'.
 5 alice execWrite: 'alice.txt' with: 'alice''s info'.
 6 bob execCreate: 'bob.txt'.
 7 bob execWrite: 'bob.txt' with: 'bob''s info'.
 8 ('Alice reads her file: ',
 9   (alice execRead: 'alice.txt')) writeLine.
10 ('Bob reads his file: ',
11   (bob execRead: 'bob.txt')) writeLine.
12 ('Alice reads Bob''s file: ',
13   (alice execRead: 'bob.txt')) writeLine
```

Листинг 3. Моделирование действий пользователей на языке AspectTalk

Объём статьи не позволяет дать полный текст программы. Отметим только, что класс Process является экземпляром метакласса ProcessClass, а класс File является экземпляром метакласса FileClass. Отдельно выделяются класс и метакласс системы (System и SystemClass соответственно).

В листинге 4 показан один из возможных вариантов определения указанных метаклассов. Такие метаклассы будем называть пустыми, поскольку они не несут никакой семантической нагрузки.

```
1 self addVariables: #(
2   #OwnerClass
3   #ThingClass
4   #SystemClass
5 ).
6 OwnerClass <- Metaclass new.
7 ThingClass <- Metaclass new.
8 SystemClass <- Metaclass new
```

Листинг 4. «Пустые» метаклассы на языке AspectTalk

Если исполнить программу в таком виде, как она описана выше, то можно увидеть, что alice имеет возможность успешно прочитать файл процесса bob.

Вывод программы на экран показан в листинге 5.

```
Alice reads her file: alice's info
Bob reads his file: bob's info
Alice reads Bob's file: bob's info
```

Листинг 5. Вывод программы, использующей «пустой» аспект

В листинге 6 показан аспект безопасности данной системы, реализованный на языке AspectTalk. Предполагается, что после объединения этого аспекта с главным аспектом программы прочитать данные, хранящиеся в каком-либо файле, сможет лишь пользователь, создавший этот файл.

В строках 1 – 7 объявляются переменные, используемые при работе алгоритма, а в строках 8 и 9 две из них – глобальные переменные, предназначенные для хранения идентификаторов, – инициализируются нулями.

В строках 10 – 18 определяется метакласс OwnerClass: его задача – в перехвате сообщения init, посылаемого каждому объекту при его создании (т.е. соответствующий алгоритм является конструктором объекта). В строке 13 выполняется вызов оригинального конструктора, а после к объекту добавляется член-данное id и инициализируется некоторым уникальным значением (хранящимся в переменной currentId).

В строках 19 – 24 определяется метакласс SystemClass: его задача – при обращении всякого объекта к нему, с целью выполнения действий над файлами, записать идентификатор этого объекта в глобальную переменную processId. Будут перехвачены только сообщения, селектор которых удовлетворяет регулярному выражению (read|write|create)File.* (строка 20). Непосредственно запись идентификатора происходит в строке 22, а в строке 23 вызывается оригинальный алгоритм для перехваченного сообщения.

```
1 self addVariables: #(
  2   #processId
  3   #currentId
  4   #OwnerClass
  5   #ThingClass
  6   #SystemClass
  7 ).
  8 processId <- 0.
  9 currentId <- 0.
 10 OwnerClass <- Metaclass new;
 11   addEnvelope: 'init' usingBlock: [
 12     | :sender :message :args |
 13     receiver call: message with: args.
 14     receiver addVariable: #id.
 15     currentId <- currentId + 1.
 16     id <- currentId.
 17     receiver addMethod: #id usingBlock: [ ^ id ]
 18   ].
 19 SystemClass <- Metaclass new;
 20   addEnvelope: '(read|write|create)File.*' usingBlock: [
 21     | :sender :message :args |
 22     processId <- sender id.
 23     receiver call: message with: args
 24   ].
 25 ThingClass <- Metaclass new;
 26   addEnvelope: 'init' usingBlock: [
 27     | :sender :message :args |
 28     receiver call: #init with: args.
 29     receiver addVariable: #id.
 30     id <- processId
 31   ];
 32   addEnvelope: '(read|write).*' usingBlock: [
 33     | :sender :message :args result |
 34     result <- 'This is NOT YOUR file!'.
 35     [ processId = id ] ifTrue: [
 36       result <- receiver call: message with: args
 37     ].
 38     ^ result
 39   ]
```

Листинг 6. Аспект безопасности, описанный на языке AspectTalk

В строках 25 – 39 определяется метакласс ThingClass: его задача – добавить идентификатор ко всякому объекту, соответствующему этому метаклассу (это производится в перехватчике метода init в строках 26 – 31). Как видно, идентификатор объекта становится равен идентификатору, хранящемуся в переменной processId, заполненной метаклассом SystemClass при обращении процесса к системе с просьбой создать файл (сообщение #createFile:).

В строках 32 – 39 определяется обработчик сообщений на чтение и на запись (соответствующее регулярное выражение (read|write).* указано в строке 32). В нём производится проверка идентификатора объекта-процесса, обратившегося к системе с просьбой о чтении или записи, и идентификатора объекта-файла. В случае совпадения идентификаторов производится запрашиваемое действие, а в случае несовпадения возвращается строка 'This is NOT YOUR file!'.

Вывод программы, полученной объединением главного аспекта и аспекта безопасности, показан в листинге 7.

```
Alice reads her file: alice's info
Bob reads his file: bob's info
Alice reads Bob's file: This is NOT YOUR file!
```

Листинг 7. Вывод программы, использующей аспект безопасности

Стоит отметить также сходство языка AspectTalk и языка MetaclassTalk, описанного в [16], несмотря на то, что их разработка производилась независимо. Так же, как и MetaclassTalk, язык AspectTalk обладает следующими преимуществами над языком AspectJ.

1. Возможность повторного использования аспектов в других приложениях.
2. Унифицированное определение статических конструкций аспекта (например, состав объектов – их член-данные, отношения наследования и агрегации) и динамических конструкций (перехват сообщений).
3. Возможность использования многих аспектов в одном приложении, пересекающихся по одному и тому же модулю.

Детальный анализ данного сравнения можно найти в [16].

Заключение

Аспектно-ориентированное программирование – одна из главных современных парадигм программирования. Она вводит уровень модульности, позволяющий изолированно разрабатывать различные аспекты приложения. Реализация политик безопасности – важный аспект современных компьютерных систем. АОП – это инструмент, позволяющий реализовывать политики безопасности удобным для человека образом. Такой инструмент в составе языка программирования AspectTalk, виртуальной машины и транслятора с AspectTalk на язык этой машины создан автором данной работы.

Грамматика языка AspectTalk включает в себя как объектно-ориентированные конструкции, которые используются при реализации основных частей приложений, так и аспектно-ориентированные конструкции, необходимые для реализации аспектов приложений. Программы, написанные на нем, после трансляции объединяются в одно целое и исполняются виртуальной машиной.

Язык AspectTalk – объектно-ориентированный и аспектно-ориентированный язык программирования, обладающий некоторыми преимуществами перед современными аспектно-ориентированными средствами разработки. Основой для AspectTalk послужил объектно-ориентированный язык программирования Smalltalk [20, 21], аспектно-ориентированное программирование реализуется с помощью метаобъектного протокола, аналогичного таковому в языке Python [15]. Язык AspectTalk схож с языком MetaclassTalk, описанным в [16], несмотря на их независимую реализацию. Оба языка обладают определёнными преимуществами над AspectJ в плане удобства реализации аспектов.

Поскольку аспектно-ориентированное программирование является перспективным способом реализации политик безопасности, исследования в области АОП языков и методов разработки способны вывести внедрение политик безопасности в компьютерные системы на качественно новый уровень.

ЛИТЕРАТУРА

1. *Department* of Defense Trusted Computer System Evaluation Criteria. DoD 5200.28-STD, 1985.
2. *Девянин П.Н.* Модели безопасности компьютерных систем: Учеб. пособие для студ. высш. учеб. заведений. М.: Издательский центр «Академия», 2005. 144 с.
3. *Таненбаум Э.* Современные операционные системы. 2-е изд. СПб.: Питер, 2005. 1038 с.
4. *Dijkstra E.W.* Selected Writings on Computing: A Personal Perspective. N.Y.: Springer Verlag, 1982. P. 60 – 66.
5. *Elrad T.*, *Aksit M.M.*, *Kiczales G.*, *et al.* Discussing Aspects of AOP // Communications of ACM. 2001. October. V. 44. No. 10. P. 33 – 38.
6. *Booch G.* Through the Looking Glass, 2001. http://www.ddj.com/architect/184414752.
7. *Elrad T.*, *Filman R.E.*, *Bader A.* Aspect-Oriented Programming // Communications of ACM. 2001. October. V. 44. No. 10. P. 29 – 32.

8. Язык программирования AspectJ. http://eclipse.org/aspectj.
9. *Diaz Pace J.A.*, *Campo M.R.* Analyzing the Role of Aspects in Software Design // Communications of ACM. 2001. October. V. 44. No. 10. P. 67 – 73.
10. *Lieberherr K.*, *Orleans D.*, *Ovlinger J.* Aspect-Oritnted Programming with Adaptive Methods // Communications of ACM. 2001. October. V. 44. No. 10. P. 39 – 41.
11. *Bergmans L.*, *Aksit M.* Composing Crosscutting Concerns Using Composition Filters // Communications of ACM. 2001. October. V. 44. No. 10. P. 51 – 57.
12. *Kiczales G.*, *Hilsdale E.*, *Hugunin J.*, *et al.* Getting Started with AspectJ // Communications of ACM. 2001. October. V. 44. No. 10. P. 59 – 65.
13. *Kiczales G.*, *Lamping J.*, *Menhdhekar A.*, *et al.* Aspect-Oriented Programming // Lecture Notes in Computer Science / Ed. by M. Aksit, S. Matsukoa. N.Y.: Springer Verlag, 1997. June. V. 1241. P. 220 – 242.
14. *Aspect*-Oriented Software Development. http://aosd.net.
15. Язык программирования Python. http://python.org.
16. *Bouraqadi N.*, *Seriai A.*, *Leblanc G.* Towards unified aspect-oriented programming // ESUG 2005 Research Conference. Brussels, Belgium, 2005. 22 p.
17. *Куликов М.Л.*, *Ромашкин Е.В.*, *Стефанцов Д.А.* Разработка средств моделирования политик безопасности операционных систем // Вестник ТГУ. Приложение. 2007. № 23. С. 189 – 193.
18. *Budd T.* An Introduction to Object-Oriented Programming. 3rd edition. Addison-Wesley, 2001. 648 p.
19. *Kiczales G.* The Art of Meta-Object Protocol. The MIT Press, 1991. 345 pp.
20. *Goldberg A.*, *Robson D.* Smalltalk 80, volume 1 – The Language and its implementation. Addison-Wesley, 1983.
21. *Budd T.* A Little Smalltalk. Addison-Wesley, 1987. 280 p.

ПРИКЛАДНАЯ ТЕОРИЯ ГРАФОВ

DOI 10.17223/20710410/1/15

УДК 519.17

СЕМЕЙСТВА ТОЧНЫХ РАСШИРЕНИЙ ТУРНИРОВ

М.Б. Абросимов, А.А. Долгов

Саратовский государственный университет

E-mail: mic@rambler.ru

В работе рассматриваются некоторые вопросы, связанные с точными расширениями турниров. Обобщается одно семейство, полученное ранее, и предлагаются два новых бесконечных семейства турниров, имеющих точные 1-расширения.

Ключевые слова: *орграф*, *турнир*, *минимальное расширение*, *точное расширение*, *оптимальная отказоустойчивая реализация*, *точная отказоустойчивая реализация.*

Ориентированным графом (далее – *орграфом*) называется пара $G = (V, \alpha)$, где V – конечное непустое множество, называемое *множеством вершин*, а α – отношение на множестве вершин V, называемое *отношением смежности.*

Граф с симметричным и антирефлексивным отношением смежности называется *неориентированным графом* (далее – *графом*). Граф с антисимметричным отношением смежности называется *направленным графом* или *диграфом*. Граф с пустым отношением смежности называется вполне несвязным и обозначается O_n, где n – число вершин. Орграф с универсальным отношением смежности называется полным и обозначается $\overrightarrow{K_n}$.

Симметризацией орграфа $\vec{G} = (V, \alpha)$ называется граф $G = (V, (\alpha \cup \alpha^{-1}) \setminus \Delta)$, то есть симметризация орграфа получается заменой дуг ребрами и удалением петель. Симметризация полного орграфа $\overrightarrow{K_n}$ называется *полным* (*неориентированным*) *графом* и обозначается K_n. Полный диграф без петель называется *турниром* и обозначается T_n. Турнир можно рассматривать как граф, получающийся из полного графа K_n приданием каждому ребру некоторой ориентации. Основные определения даются в соответствии с работой [1].

Вложением графа $G_1 = (V_1, \alpha_1)$ в граф $G_2 = (V_2, \alpha_2)$ называется такое взаимно однозначное отображение $f : V_1 \to V_2$, что для любых вершин u, $v \in V_1$ выполняется следующее условие: $(u, v) \in \alpha_1 \Rightarrow (f(u), f(v)) \in \alpha_2$.

Два графа $G_1 = (V_1, \alpha_1)$ и $G_2 = (V_2, \alpha_2)$ называются *изоморфными*, если можно установить взаимно однозначное соответствие $f : V_1 \to V_2$, сохраняющее отношение смежности: $(u, v) \in \alpha_1 \Leftrightarrow (f(u), f(v)) \in \alpha_2$, для любых u, $v \in V_1$. В этом случае пишут $G_1 \cong G_2$.

Граф $G^* = (V^*, \alpha^*)$ называется *минимальным k-расширением* (k – натуральное) n-вершинного графа $G = (V, \alpha)$, если выполняются следующие условия:

1. G^* является k-расширением G, то есть граф G вложим в каждый подграф графа G^*, получающийся удалением любых его k вершин.
2. G^* содержит $n+k$ вершин, то есть $|V^*| = |V| + k$.
3. α^* имеет минимальную мощность при выполнении условий 1 и 2.

Граф G_t^* называется *точным k-расширением* графа G, если любой граф, получающийся удалением произвольных k вершин графа G_t^*, изоморфен графу G.

Понятие минимального k-расширения введено на основе понятия оптимальной k-отказоустойчивой реализации, которое было предложено Хейзом в работе [2]. Понятие точного расширения впервые было введено Харари и Хейзом в работе [3], где рассматривались неориентированные графы. Там же были предложены некоторые семейства графов, которые имеют точное расширение. В работе [4] удалось установить, что среди неориентированных графов только полный и вполне несвязный графы имеют точные k-расширения при любом k. Среди остальных графов выделяются графы, имеющие степенное множество вида $\{b, b-1\}$, у которых число вершин степени $b-1$ в точности равно b. Такие графы могут иметь точное 1-расширение, но не могут иметь точное k-расширение ни при каком $k > 1$. В работах [5, 6] удалось установить связь точных k-

расширений орграфов с точными *k*-расширениями неориентированных графов и предложить два семейства точных расширений турниров. Данная работа является продолжением в исследовании семейств точных расширений турниров. Приведем некоторые результаты, полученные ранее в [5, 6].

Теорема 1. Пусть $\overrightarrow{G^*} = (V^*, \alpha^*)$ – точное *k*-расширение орграфа $\vec{G} = (V, \alpha)$. Тогда отношения смежности α и α^* являются одновременно либо рефлексивными, либо антирефлексивными.

Теорема 2. Пусть $\overrightarrow{G^*}$ – точное *k*-расширение орграфа $\vec{G}$. Тогда симметризация $\overrightarrow{G^*}$ является точным *k*-расширением симметризации $\vec{G}$.

Теорема 3. Пусть *G* – дграф с числом вершин, большим 1, тогда его точное *k*-расширение, если оно есть, также будет дграфом.

Теорема 4. Пусть $\overrightarrow{G^*}$ – точное *k*-расширение орграфа $\vec{G}$. Тогда дополнение $\overrightarrow{G^*}$ является точным *k*-расширением дополнения $\vec{G}$.

Из приведенных теорем следует, что точное *k*-расширение турнира, если оно существует, также является турниром. Причем дополнение турнира (а это тоже будет турнир) также будет иметь точное *k*-расширение.

Изоморфизм графа на самого себя называется *автоморфизмом*. Две вершины *u* и *v* графа *G* называются *подобными*, если существует автоморфизм графа *G*, при котором образом вершины *u* является вершина *v*. Граф, все вершины которого подобны, называется *вершинно-симметрическим*. Граф, не имеющий ни одной пары подобных вершин, называется *асимметричным*.

Для неориентированных графов в работе [7] было доказано, что граф тогда и только тогда является точным 1-расширением, когда он является вершинно-симметрическим. Оказывается, что для орграфов в общем случае подобное утверждение неверно. Далее будут указаны турниры, которые не являются вершинно-симметрическими графами, но являются точными 1-расширениями. Связь точных расширений с вершинно-симметрическими орграфами может быть уточнена следующим образом (см. [6]):

Теорема 5. Всякий вершинно-симметрический орграф является точным 1-расширением подходящего орграфа.

1. Семейство транзитивных турниров (T)

Первое из рассматриваемых в данной работе семейств точных расширений турниров – семейство транзитивных турниров – было впервые описано в работе [5]. *Транзитивный турнир* – это турнир, у которого из существования дуг (*u*, *v*) и (*v*, *w*) вытекает существование дуги (*u*, *w*). В частности, в работе [5] доказывается следующая

Теорема 6. Единственным минимальным *k*-расширением транзитивного турнира T_n при $n > 2$ является транзитивный турнир T_{n+k}. Причем это расширение является и его точным *k*-расширением.

При $n = 2$ транзитивный турнир T_2 имеет два точных 1-расширения – циклическую и транзитивную тройки. На рис. 1 указана схема построения транзитивного турнира.

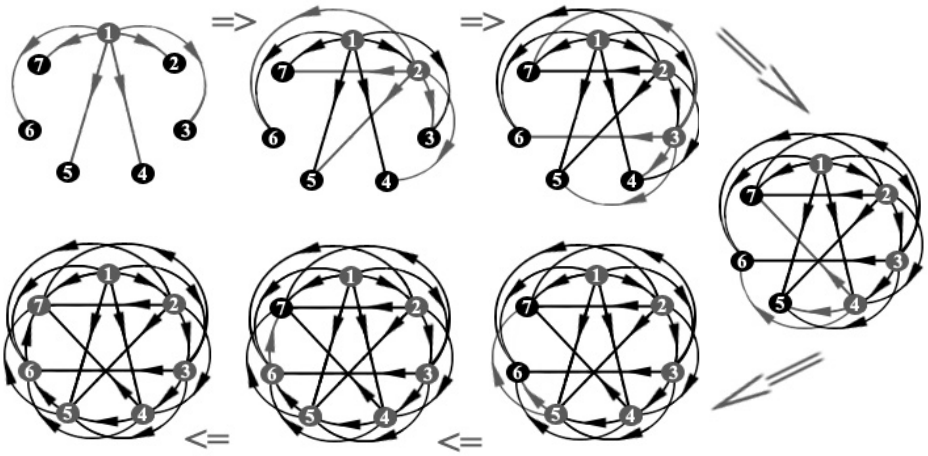


Рис. 1

Как видно из схемы, для построения *n*-вершинного транзитивного турнира необходимо каждую вершину v_i соединить с вершинами v_{i+1}, v_{i+2}, …, v_n. Очевидно, что для любого числа вершин существует в точности

один транзитивный турнир. Транзитивный турнир является *самодополнительным* графом, то есть он изоморфен своему дополнению. На основе транзитивных турниров удалось предложить новое семейство – семейство полутранзитивных турниров, описываемое далее.

Заметим, что в транзитивном турнире все вершины имеют различные полустепени исхода и захода, и следовательно, в нем нет ни одной пары подобных вершин, то есть транзитивный турнир является асимметрическим графом. Таким образом, для орграфов могут существовать точные k-расширения, являющиеся ассиметричными графами. Семейство транзитивных турниров – первое и пока единственное, найденное из таких семейств.

2. Семейство вершинно-симметрических турниров (S)

В работе [6] предлагается схема построения семейства вершинно-симметрических турниров (см. рис. 2).

Рассмотрим полный $(2n + 1)$-вершинный граф K_{2n+1}. Разместим его вершины в вершинах правильного n-угольника. Ориентируем ребра K_{2n+1} следующим образом. Для произвольной вершины v и n следующих за ней по часовой стрелке вершин ребра ориентируем в направлении от v, а для оставшихся n вершин, расположенных от v против часовой стрелки, ориентируем ребра в направлении к v. После проведенной ориентации все вершины будут иметь степени исхода и захода равными n.

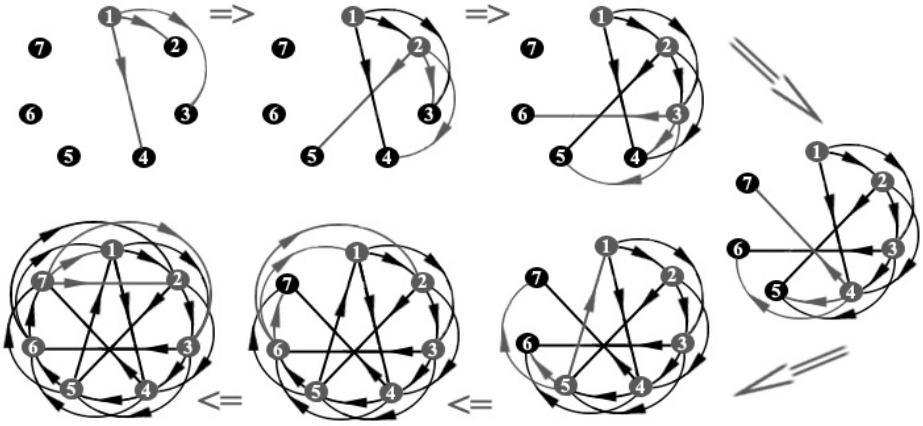


Рис. 2

Очевидно, что построенный по описанной схеме турнир будет являться вершинно-симметрическим, более того, он обладает *циклической симметрией*. По теореме 5 любой вершинно-симметрический граф является точным 1-расширением некоторого подходящего графа. Однако предложенная схема не перечисляет всех вершинно-симметрических турниров.

Рассмотрим алгоритм построения вершинно-симметрических турниров с циклической симметрией.

Пусть G – произвольный вершинно-симметрический граф. Все вершины у такого графа подобны. Это означает, что если такой граф обладает циклической симметрией, то, расположив вершины по кругу, дуги можно нарисовать так, что при повороте графа на угол $360/n$ в любую сторону будет все время получаться одно и то же изображение. Матрица смежности изображенного таким образом графа будет обладать интересным свойством: каждая строка матрицы будет получаться из предыдущей строки циклическим сдвигом на одну позицию вправо. Таким образом, чтобы перебрать все вершинно-симметрические графы с циклической симметрией, необходимо перебрать все возможные первые строки матрицы смежности и достроить последующие по указанному алгоритму. Однако таким образом могут получиться не только турниры.

На главной диагонали матрицы смежности турнира всегда будут нули. Попробуем разместить в первой позиции единицу и достроить матрицу по указанному алгоритму:

0 1 x x x x **0**
0 0 1 x x x x
x **0** 0 1 x x x
x x **0** 0 1 x x
x x x **0** 0 1 x
x x x x **0** 0 1
1 x x x x **0** 0

Поскольку у турнира нет встречных дуг, то на последней позиции первой строки необходимо поставить ноль. Таким образом, в случае турниров половина первой строки матрицы смежности при использовании предложенного алгоритма однозначно определяет вторую половину.

Для построения всех n-вершинных вершинно-симметрических турниров с циклической симметрией необходимо перебирать не всю первую строку матрицы смежности, а только половину, вторая половина получается из первой, если ее перевернуть и инвертировать, то есть заменить 0 на 1, а 1 на 0.

Очевидно, что вершинно-симметрические турниры существуют только при нечетном числе вершин. Дополнение турнира с циклической симметрией также является турниром с циклической симметрией и таким образом будет построено по этой же схеме.

3. Семейство турниров точных 1-расширений с четным числом вершин (E1)

На рис. 3 приведена схема построения семейства турниров с числом вершин вида $n = 4*k+2$. Для построения необходимо расположить вершины по кругу. Далее проводим дуги из каждой нечетной вершины в $n/2$ последующих, а из каждой четной в $n/2 - 1$ последующих вершин.

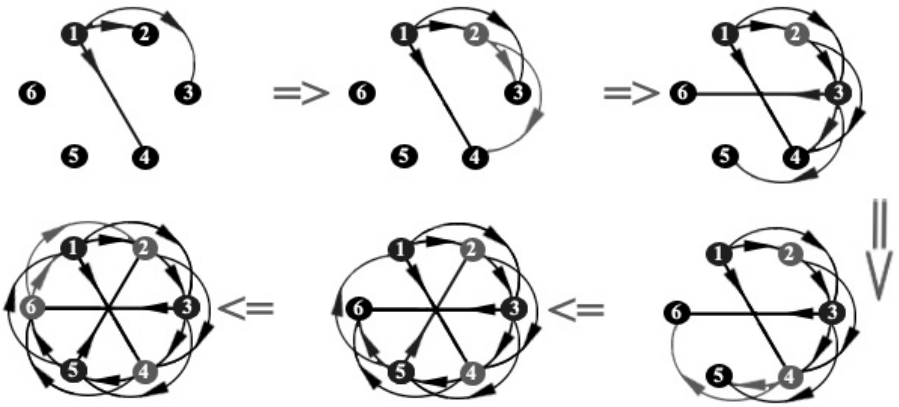


Рис. 3

Теорема 7. Турниры с числом вершин вида $n = 4*k+2$, построенные по указанной схеме, являются точными 1-расширениями.

Доказательство. Покажем, что схема действительно позволяет строить точные 1-расширения турниров. По построению имеем две группы подобных вершин. На рис. 3 это вершины с четными и нечетными метками. Каждая вершина с нечетным номером имеет степень исхода $n/2$, каждая вершина с четным номером – степень исхода $n/2–1$.

Разобьем все множество вершин на пары: [1, 2] [3, 4] [5, 6], …

Рассмотрим произвольную пару $[v_i, v_i+1]$:

Вершина v_i соединена с $n/2$ вершинами v_i+1, v_i+2, …, $v_i+n/2$.

Вершина v_i+1 соединена с $n/2–1$ вершиной v_i+2, …, $v_i+n/2$.

Таким образом, у каждой пары совпадают все дуги, кроме одной, которая находится между ними.

Так как $n = 4*k+2$, то $n/2$ будет нечетным числом, $n/2–1$ четным, таким образом, дуги исходящие из произвольной пары, захватывают другие пары целиком, не разбивая их. В обобщенном виде это утверждение можно записать так: $[v_i, v_i+1] => [v_i+2, v_i+3], [v_i+4, v_i+5], [v_i+6, v_i+7] ... [v_i+n/2–1, v_i+n/2]$.

Из всего вышесказанного легко заметить, что графы, полученные при удалении одной вершины пары, будут изоморфны. Действительно, рассмотрим произвольную пару: в нее входят дуги из $(n–2)/4$ предыдущих пар, и из нее выходят дуги в $(n–2)/4$ последующих пар. При удалении одной из вершин пары получаем вершину, в которую входят дуги из тех же $n/4$ пар и выходят дуги в те же $n/4$ пары.

Так как все вершины с четными и нечетными метками подобны и граф, получающийся при удалении некоторой вершины с четной меткой, изоморфен графу, получающемуся при удалении некоторой вершины с нечетной меткой, то все подграфы любого графа семейства будут изоморфны. ■

Заметим, что представители данного семейства не являются вершинно-симметрическими графами.

4. Семейство полутранзитивных турниров (H)

Известно, что у транзитивного турнира (как и у любого бесконтурного графа) существует правильная нумерацию вершин, при которой дуга идет только из вершины с меньшим номером в вершину с большим. При правильной нумерации вершин матрица смежности транзитивного турнира станет верхней треугольной

матрицей. Полутранзитивный турнир получается из транзитивного турнира с числом вершин $n = 3*k$ по схеме, указанной на рис. 4.

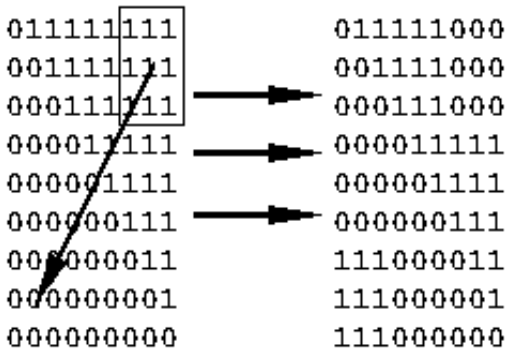


Рис. 4

Как видно из схемы, для построения полутранзитивного турнира необходимо в матрице смежности транзитивного турнира переориентировать правый верхний ее квадрат $n/3 \times n/3$.

Теорема 8. Каждый полутранзитивный турнир является точным 1-расширением подходящего турнира.

Доказательство. Пусть T – полутранзитивный турнир с числом вершин $n = 3*k$, матрица смежности которого получена по схеме, изображенной на рис. 4. Обозначим его вершины через $v_1, v_2, \ldots, v_n$.

Если удалять по одной вершине из турнира T, то будут получены три группы матриц смежности определенного вида (рис. 5), а все множество вершин будет разбито на три класса подобных вершин.

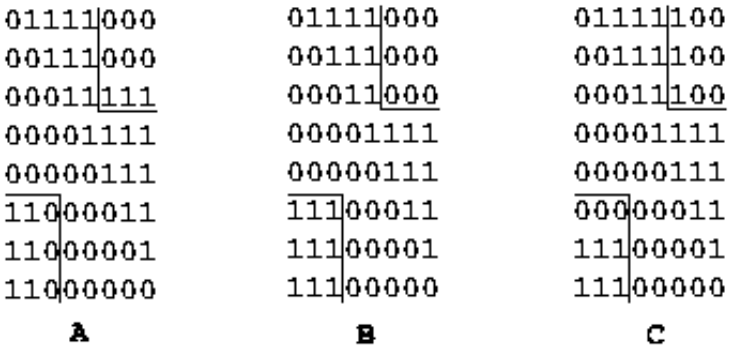


Рис. 5

Очевидно, что для того, чтобы из матрицы вида A получить матрицу вида C, необходимо часть вершин, расположенных в начале, перенести в конец по следующей схеме: $v_1, v_2, v_3, v_4, \ldots v_n \Rightarrow v_{n/3}, v_{n/3+1}, \ldots, v_n, v_{n/3-1}, v_{n/3-2}, \ldots, v_1$ (рис. 6).

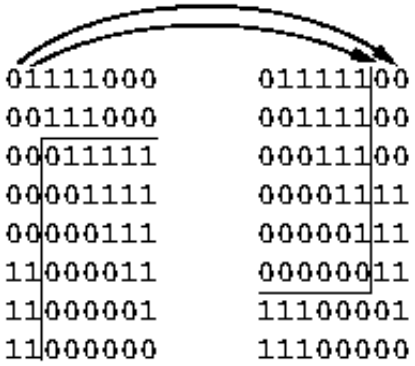


Рис. 6

Если продолжить процесс и переставить еще $n/3$ вершин по заданной схеме в матрице вида C, то получится матрица вида B (рис. 7).

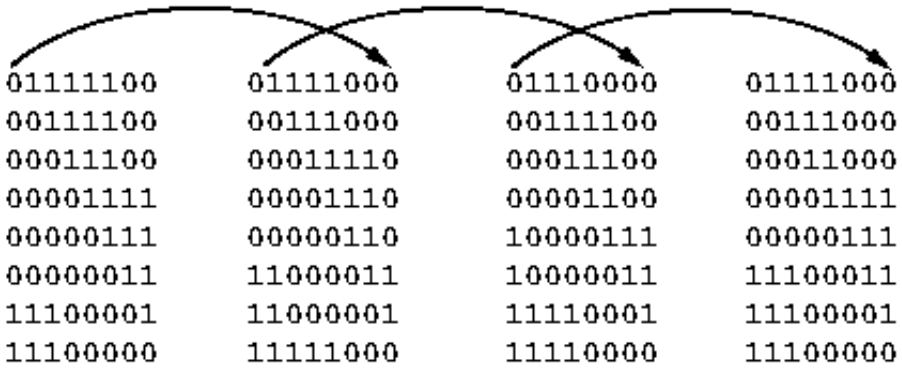


Рис. 7

Таким образом, все максимальные подграфы полутранзитивного турнира изоморфны. ■

Далее в таблице указываются все турниры с числом вершин до 10 и их точные 1-расширения, а также семейства, к которым относятся эти точные 1-расширения. Турниры в таблице представлены своими максимальными минорными кодами (см. [6]) и векторами, составленными из степеней исхода вершин в порядке неубывания. В 1-м и 4-м столбцах указывается число вершин турнира и его точного 1-расширения соответственно.

В приведенной таблице только два турнира не принадлежат ни одному из описанных семейств: 5-вершинный турнир с кодом 982 и 9-вершинный турнир с кодом 68424641622. Дополнительный интерес эти турниры представляют тем, что они имеют точные 1- и 2-расширения, не имея точных k-расширений при $k > 2$. Такая ситуация невозможна для неориентированных графов, а для орграфов это пока единственные известные примеры.

N	Максимальный минорный код турнира	Вектор степеней турнира	N	Семейство	Максимальный минорный код точного 1-расширения	Вектор степеней точного расширения
2	1	(0, 1)	**3**	S, H	5	(1, 1, 1)
2	1	(0, 1)	**3**	T	7	(0, 1, 2)
3	7	(0, 1, 2)	**4**	T	63	(0, 1, 2, 3)
4	59	(1, 1, 2, 2)	**5**	S	947	(2, 2, 2, 2, 2)
4	63	(0, 1, 2, 3)	**5**	T	1023	(0, 1, 2, 3, 4)
5	**982**	**(1, 2, 2, 2, 3)**	**6**		**31435**	**(2, 2, 2, 3, 3, 3)**
5	1011	(1, 2, 2, 2, 3)	**6**	E1, H	32359	(2, 2, 2, 3, 3, 3)
5	1023	(0, 1, 2, 3, 4)	**6**	T	32767	(0, 1, 2, 3, 4, 5)
6	**31435**	**(2, 2, 2, 3, 3, 3)**	**7**	S	**2011853**	**(3, 3, 3, 3, 3, 3, 3)**
6	32487	(2, 2, 2, 3, 3, 3)	**7**	S	2079175	(3, 3, 3, 3, 3, 3, 3)
6	32767	(0, 1, 2, 3, 4, 5)	**7**	T	2097151	(0, 1, 2, 3, 4, 5, 6)
7	2097151	(0, 1, 2, 3, 4, 5, 6)	**8**	T	268435455	(0, 1, 2, 3, 4, 5, 6, 7)
8	267242391	(3, 3, 3, 3, 4, 4, 4, 4)	**9**	S	68414052123	(4, 4, 4, 4, 4, 4, 4, 4, 4)
8	267716381	(3, 3, 3, 3, 4, 4, 4, 4)	**9**	S	68535393593	(4, 4, 4, 4, 4, 4, 4, 4, 4)
8	268298127	(3, 3, 3, 3, 4, 4, 4, 4)	**9**	S	68684320527	(4, 4, 4, 4, 4, 4, 4, 4, 4)
8	268428175	(2, 3, 3, 3, 4, 4, 4, 5)	**9**	H	68717612831	(3, 3, 3, 4, 4, 4, 5, 5, 5)
8	268435455	(0, 1, 2, 3, 4, 5, 6, 7)	**9**	T	68719476735	(0, 1, 2, 3, 4, 5, 6, 7, 8)
9	**68424641622**	**(3, 3, 4, 4, 4, 4, 4, 5, 5)**	**10**		**35033416510570**	**(4, 4, 4, 4, 4, 5, 5, 5, 5, 5)**
9	68717879055	(3, 3, 4, 4, 4, 4, 4, 5, 5)	**10**	E1	35183554076191	(4, 4, 4, 4, 4, 5, 5, 5, 5, 5)
9	68719476735	(0, 1, 2, 3, 4, 5, 6, 7, 8)	**10**	T	35184372088831	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
10	**35033416510570**	**(4, 4, 4, 4, 4, 5, 5, 5, 5, 5)**	**11**	S	**35874218506823866**	**(5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5)**
10	35183822519839	(4, 4, 4, 4, 4, 5, 5, 5, 5, 5)	**11**	S	36028234260315167	(5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5)
10	35105255749749	(4, 4, 4, 4, 4, 5, 5, 5, 5, 5)	**11**	S	35947781887743157	(5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5)
10	35175217913391	(4, 4, 4, 4, 4, 5, 5, 5, 5, 5)	**11**	S	36019423143312439	(5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5)
10	35184372088831	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9)	**11**	T	36028797018963967	(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

ЛИТЕРАТУРА

1. *Богомолов А.М.*, *Салий В.Н.* Алгебраические основы теории дискретных систем. М.: Наука, 1997.
2. *Hayes J.P.* A graph model for fault-tolerant computing system // IEEE Trans. Comput. 1976. V. C-25. No. 9. P.875 – 884.
3. *Harary F.*, *Hayes J.P.* Node fault tolerance in graphs // Networks. 1996. V. 27. P. 19 – 23.
4. *Абросимов М.Б.* Минимальные расширения дополнений графов // Теоретические задачи информатики и ее приложений. Саратов: СГУ, 2001. Вып. 4. С. 11 – 19.
5. *Абросимов М.Б.* Минимальные расширения транзитивных турниров // Вестник ТГУ. Приложение. 2006. № 17. С. 187 – 190.
6. *Абросимов М.Б.*, *Долгов А.А.* Точные расширения некоторых турниров // Вестник ТГУ. Приложение. 2007. № 23. С. 211 – 216.
7. *Абросимов М.Б.* Некоторые вопросы о минимальных расширениях графов // Изв. Саратовского университета. 2006. № 6. С. 86 – 91.

2008 Прикладная теория графов № 1(1)

DOI 10.17223/20710410/1/16

УДК 512.1, 519.6, 519.7

ИНСТРУМЕНТАРИЙ ГРАФИЧЕСКОГО ИССЛЕДОВАНИЯ СИМВОЛЬНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ[1]

А.А. Евдокимов, А.А. Левин

Институт математики им. С.Л. Соболева СО РАН, г. Новосибирск

E-mail: evdok@math.nsc.ru, levin@math.nsc.ru

Разработан пакет «BruijnViz» для исследования свойств символьных последовательностей, или слов большой длины. Все подслова длины *n* отображаются на граф перекрытия слов (граф де Брёйна), образуя граф-портреты в процессе роста длины последовательности. Реализованы различные способы изображения графа на плоскости экрана. Приводятся примеры граф-портретов последовательностей, возникающих в приложениях, и анализируются их свойства.

Ключевые слова: *граф де Брёйна*, *граф подслов*, *символьная последовательность*, *сложность*, *визуализация графа.*

Вначале об идее подхода к задаче визуализации символьных последовательностей.

При исследовании символьных последовательностей эффективным оказывается анализ связи их свойств со свойствами структур, образуемых множеством фрагментов (подслов) этой последовательности. Выявление таких структур и исследование динамики их изменения при увеличении длины фрагментов и самой последовательности дает ценную информацию о ее свойствах. Возможности изучения свойств последовательностей зависят от «хорошего» изображения их графов перекрытия подслов на плоскости (на экране компьютера).

Графы перекрытия слов, введенные де Брёйном в 1946 г. и теперь называемые его именем [1], оказываются удобными для изображения последовательностей и изучения их структурных свойств. Изображение графа подслов последовательности на графе де Брёйна мы называем граф-портретом этой последовательности. Рассмотрение динамики последовательности граф-портретов помогает исследовать свойства как отдельных последовательностей, так и их классов.

Прикладная направленность исследования граф-портретов последовательностей состоит в расширении методов и инструментария для анализа структуры последовательностей как естественного происхождения, например генетических [3, 4], так и математических – порождаемых рекурсивными процедурами различного типа. Анализ граф-портретов символьных последовательностей выявил ряд их интересных свойств, связанных с особенностями структуры множества подслов. В частности, с помощью пакета «BruijnViz» были исследованы последовательность, введенная Евдокимовым для решения известной проблемы «змея в ящике», и последовательность непрерывного кодирования, близкая к последовательности, известной в англоязычной литературе как последовательность «Look and say» [5].

Наиболее интересные полученные граф-портреты демонстрируются в работе, в частности последовательность непрерывного кодирования имеет сложные граф-портреты и большую комбинаторную сложность (количество различных её подслов).

Задача построения граф-портретов последовательностей приводит к задачам поиска вложений графов, сохраняющих определенные структурные свойства вкладываемых объектов: метрические, алгебраические или комбинаторные [6 – 8], в частности построения таких вложений графов на плоскость, которые сохраняют отношение близости между вершинами, а расстояния между далекими вершинами оставляют больше некоторого заданного порога [7]. Идея такого типа вложений для задач визуализации граф-портретов символьных последовательностей реализована в работе [9].

1. Определения

Теперь определим основные понятия. Вершинами графа де Брёйна B_m^n размерности *n* являются всевозможные слова длины *n* в алфавите из *m* букв. Две вершины $\tilde{\alpha} = (\alpha_1, ..., \alpha_n)$ и $\tilde{\beta} = (\beta_1, ..., \beta_n)$ соединены дугой,

[1] Исследование выполнено при финансовой поддержке РФФИ (проект 08-01-00671) и программы Отделения математических наук РАН «Алгебраические и комбинаторные методы математической кибернетики» (проект «Новые методы дискретного анализа и комбинаторной оптимизации»).

ориентированной от $\tilde{\alpha}$ к $\tilde{\beta}$, тогда и только тогда, когда $\alpha_2 = \beta_1$, $\alpha_3 = \beta_2$, ... , $\alpha_n = \beta_{n-1}$, т.е. когда слова $\tilde{\alpha}$ и $\tilde{\beta}$ перекрываются по $n - 1$ буквам.

Граф B_m^n имеет m петель в вершинах, соответствующих словам – константам, состоящим из одной буквы алфавита. Он связен, однороден, полустепень входа и выхода каждой его вершины равна m. При изображении последовательности графов B_m^n на плоскости для $n = 1, 2, 3, \ldots$ можно использовать процедуру их построения индукцией по размерности n, основанную на том, что граф B_m^{n+1} является рёберным графом для B_m^n.

Произвольной (бесконечной или конечной длины $\geq n$) последовательности $\tilde{x} = x_1, x_2, x_3, \ldots$ букв m-алфавита сопоставляется путь в графе B_m^n, который начинается в вершине $(x_1, \ldots, x_n)$ и последовательно проходит вершины $(x_i, \ldots, x_{i+n-1})$ при $i = 2, 3, \ldots$ Заметаемый этим путем подграф графа B_m^n называется графом n-подслов последовательности $\tilde{x}$ или граф-портретом размерности n и обозначается $G^n(\tilde{x})$. Таким образом, множеством вершин $V^n(\tilde{x})$ графа $G^n(\tilde{x})$ является множество всех подслов длины n в $\tilde{x}$, а множеством дуг $E^n(\tilde{x})$ – множество всех подслов длины $n + 1$ в $\tilde{x}$ [6]. Изображение графа-дополнения $B_m^n \setminus G^n(\tilde{x})$ позволяет наблюдать структуру множества отсутствующих n-подслов в последовательности $\tilde{x}$. Для построения последовательности граф-портретов $\{G^i(\tilde{x})\}$, $i = 1, 2, 3, \ldots$, при росте их размерности $i \to i + 1$ используется операция построения рёберного графа, поскольку $G^{i+1}(\tilde{x})$ является подграфом графа рёберного для $G^i(\tilde{x})$.

2. Пакет BruijnViz

Дадим описание пакета BruijnViz и технологии его использования.

Так как все известные пакеты изображения графов ориентированы для изображения графов определенного типа, то для изображения графов подслов и исследования различных последовательностей нами разработана специальная программа BruijnViz. Программа реализована на языке JAVA, поэтому может функционировать на любой ЭВМ с виртуальной машиной JAVA. Демонстрационная версия начальной конфигурации программы находится на сервере Института математики СО РАН (http://www.math.nsc.ru/LBRT/k3/Graph/Bruijn.htm).

Программа BruijnViz строит граф B_m^n перекрытия слов (граф де Брёйна) для заданных параметров: значности алфавита последовательности m и длины слов n (размерности графа). Затем на графе изображается исследуемая последовательность $\tilde{x}$ и её граф-портрет $G^n(\tilde{x})$. В программе можно варьировать изображение на экране граф-портретов $G^n(\tilde{x})$. Изменение параметров процесса возможно производить непосредственно в ходе наблюдения.

На экране располагаются кнопки управления и меню установки режима работы программы. В верхней части экрана выводится отрезок обрабатываемой последовательности, на котором выделено текущее слово и несколько строк текущей информации. Пользователь может изменять взаимное расположение вершин графа на экране, перемещать весь граф, а также изменять размер изображения графа или его части. Если при запуске программы не задано имя файла с начальным графом, то производится построение полного графа B_m^n со случайным размещением вершин на экране. В противном случае начальный граф считывается из заданного файла и изображается на экране.

После нажатия одной из кнопок управления движением программа считывает очередной символ (в начальной точке последовательности считывается целое слово), формирует очередное слово, приписывая считанный символ в конец предыдущего слова. Вершина, имя которой совпадает с полученным словом, и ребро, соединяющее её с предыдущей вершиной, помечаются и заносятся в пройденную цепочку, а их счетчики увеличиваются на 1. Пройденная цепочка выделяется на изображении и называется «змея». Программа может непрерывно наращивать длину обработанной последовательности в широком диапазоне выбираемых скоростей. В программе реализованы режимы автоматического расталкивания близких вершин и выделения цепей на граф-портрете.

Программа позволяет изменять размерность графа де Брёйна, на котором располагается исследуемая последовательность. Увеличение размерности осуществляется построением реберного графа для всего графа

B_m^n или для части графа подслов $G^i(\tilde{x})$, пройденной последовательностью. При уменьшении размерности происходит возвращение к тому графу, из которого строился реберный. Если комбинаторная сложность последовательности растет медленно, то наблюдать граф-портреты можно для больших значений их размерности *n*, что существенно помогает при анализе свойств.

ЛИТЕРАТУРА

1. *De Bruijn N.G.* A combinatorial problem // Nederl. Akad. Wetensch. Proc. 1946. V. 49. No. 7. P. 758 – 764. (Перевод см. Кибернетический сборник, новая серия, вып. 6. М.: Мир, 1969. С. 33 – 40.)
2. *Математические* методы для анализа последовательностей ДНК: Пер. с англ. / Под ред. М.С. Уотермена. М.: Мир, 1999. 349 с.
3. *Evdokimov A.A.*, *Levin A.A.* Subwords graphs, generated by genetic sequences // Proceedings of the Third International Conference on Bioinformatics of Genome Regulation and Structure – BGRS' 2002. V. 1, IC&G. Novosibirsk, 2002. P. 131 – 133.
4. *Евдокимов А.А.*, *Левин А.А.* Графические модели и комбинаторика генетических и математических символьных последовательностей // Вычислительные технологии. 2002. Т. 7. С. 274 – 278.
5. *Евдокимов А.А.*, *Левин А.А.* Теоретическое и экспериментальное исследование рекурсивно порожденных символьных последовательностей // Вестник ТГУ. Приложение. 2007. № 23. С. 16 – 23.
6. *Евдокимов А.А.* Исследование полноты множеств слов и языков с запретами // Вестник ТГУ. Приложение. 2004. № 9(1). С. 8 – 12.
7. *Евдокимов А.А.* Кодирование структурированной информации и вложения дискретных пространств // Дискрет. анализ и исслед. операций. Сер. 1. 2000. Т. 7. № 4. С. 48 – 58.
8. *Евдокимов А.А.* Анализ, сложность и реконструкция символьных последовательностей // Вестник ТГУ. Приложение. 2005. № 14. С. 4 – 12.
9. *Евдокимов А.А.*, *Левин А.А.* Методы визуализации графов подслов символьных последовательностей // Вычислительные технологии. 2003. Т. 8. С. 5 – 11.

2008 Прикладная теория графов № 1(1)
DOI 10.17223/20710410/1/17
УДК 681.3 + 519.21

ПРЕОБРАЗОВАНИЕ РЕШЁТОК В 1-ОТКАЗОУСТОЙЧИВЫЕ ГРАФЫ

И.А. Науменко, В.Г. Скобелев

Донецкий национальный технический университет
Институт прикладной математики и механики НАН Украины, г. Донецк

E-mail: e-one@bk.ru , skbv@iamm.ac.donetsk.ua

Решается задача преобразования трансляцией по циклической группе графа некоторых n-мерных решёток в 1-отказоустойчивые графы. Оценен ряд характеристик результирующих графов.

Ключевые слова: *графы*, 1-*отказоустойчивые системы.*

В работе исследуется задача преобразования n-мерных решёток в 1-отказоустойчивые графы методом, предложенным в [1]. Детально этот метод изложен в [2, 3]. Предполагается, что в преобразуемом графе существует гамильтонов путь либо цикл [4]. При их отсутствии исходный граф преобразуется в граф, имеющий гамильтонов путь. Реконфигурация при отказах производится преобразованием отказавшей вершины (связи) на избыточную с соответствующим переименованием вершин графа по одной из следующих групп автоморфизмов: циклической группе вращений (поворот на угол $\frac{2\pi}{n}i$ $(i = 0, \ldots, n-1)$) или диэдральной группе симметрий.

Структура работы следующая: в п. 1 даны основные понятия и определения; в п. 2 исследуется преобразование простых прямоугольных n-мерных решеток, а в п. 3 – диагональных прямоугольных n-мерных решеток. Заключение содержит ряд выводов.

1. Основные понятия

Исходный объект – граф $G = (V, E)$, где V – множество вершин, E ($E \subseteq V^{(2)}$) – множество рёбер. Задачей исследования является построение такого графа $G^* = (V_1, E_1)$, что при удалении одной его вершины либо ребра редуцированный граф $G^*_{\text{ред}}$ содержит подграф $G^{\#}$, изоморфный G. При этом естественно строить не избыточный граф G^*, т.е. граф, который теряет свойство «быть 1-отказоустойчивым» при удалении хотя бы одной вершины либо ребра. Особый интерес представляют минимальные 1-отказоустойчивые графы, т.е. которые содержат наименьшее число вершин и ребер. В 1-отказоустойчивом графе G^* существуют ребра 2-х типов: рабочие и избыточные, т.е. те, которые появились в результате преобразования графа к виду 1-отказоустойчивого графа G^*. Назовем степенью вершинной избыточности графа G число $\nu_{\text{в}}(G) = |V_1|\cdot|V|^{-1}$, а степенью рёберной избыточности графа G – число $\nu_{\text{р}}(G) = |E_1|\cdot|E|^{-1}$. В дальнейшем считаем, что $V = \mathbf{Z}_k$, и, следуя [3], сохраним термин «ребро» только за такими элементами $\{z_i, z_j\} \in E$, что $| z_i - z_j | = 1$, а элемент $\{z_i, z_j\} \in E$, для которого $| z_i - z_j | > 1$, будем называть хордой длины l. Пусть $S_l(G)$ $(l \geq 2)$ – число хорд длины l в графе G, а $S(G)$ – общее число хорд в графе G, т.е.

$$S(G) = \sum_{l \geq 2} S_l(G). \qquad (1)$$

Пусть $S^{\text{раб}}(G^*)$ и $S^{\text{изб}}(G^*)$ – соответственно число рабочих и избыточных хорд в графе G^*. Тогда $S^{\text{раб}}(G^*) = S(G)$ и $S^{\text{изб}}(G^*) = S(G^*) - S^{\text{раб}}(G^*)$. Пусть $E(G)$ – число рёбер в графе G, а $E^{\text{раб}}(G^*)$ и $E^{\text{изб}}(G^*)$ – соответственно число рабочих и избыточных рёбер в графе G^*. Тогда $E^{\text{раб}}(G^*) = E(G)$ и $E^{\text{изб}}(G^*) = E(G^*) - E^{\text{раб}}(G^*)$. Характеристиками избыточности при переходе от графа G к графу G^* являются параметры $\nu_в(G)$, $\nu_p(G)$, $S^{\text{изб}}(G^*)$, $S^{\text{раб}}(G^*)$, $E^{\text{изб}}(G^*)$, $E^{\text{раб}}(G^*)$.

Рассмотрим характеристики 1-отказоустойчивых графов на примере некоторых однородных структур.

2. Простые прямоугольные n-мерные решетки

Назовем простой прямоугольной n-мерной решёткой такой граф $R_{k_1,\ldots,k_n} = (V, E)$, что $V = \overset{n}{\underset{i=1}{\times}} Z_{k_i}$ и

$$E = \{\{(v_1,\ldots, v_n), (v_1',\ldots,v_n')\} \subseteq V \mid (\exists!\, j \in \{1,\ldots,n\})(| v_j - v_j' | = 1 \;\&\; (\forall j' \neq j)(v_j = v_j'))\}.$$

Занумеруем вершины графа $R_{k_1,\ldots,k_n}$ так, что номером вершины $\boldsymbol{v} = (v_1, \ldots, v_n)$ является число

$$q(\boldsymbol{v}) = v_1 + k_1\cdot v_2 + k_1\cdot k_2\cdot v_3 + \ldots + k_1\cdot\ldots\cdot k_{(n-1)}\cdot v_n = \sum_{i=1}^{n}\left(v_i \cdot \prod_{j=0}^{n-1} k_j \right). \qquad (2)$$

Пусть $\{\boldsymbol{v}, \boldsymbol{v}'\} \in E$, где $\boldsymbol{v} = (v_1, \ldots, v_i, \ldots, v_n) \in V$ и $\boldsymbol{v}' = (v_1, \ldots, v_i + 1, \ldots, v_n) \in V$. Из (2) вытекает, что

$$q(\boldsymbol{v}) = v_1 + k_1 \cdot v_2 + \ldots + k_1 \cdot \ldots \cdot k_{i-1} \cdot v_i + \ldots + k_1 \cdot \ldots \cdot k_{(n-1)} \cdot v_n; \quad (3)$$

$$q(\boldsymbol{v}') = v_1 + k_1 \cdot v_2 + \ldots + k_1 \cdot \ldots \cdot k_{i-1} \cdot (v_i + 1) + \ldots + k_1 \cdot \ldots \cdot k_{(n-1)} \cdot v_n. \quad (4)$$

Из (3) и (4) вытекает, что

$$| q(\boldsymbol{v}) - q(\boldsymbol{v}') | = k_1 \cdot \ldots \cdot k_{i-1} \ (i = 2, \ldots, n). \quad (5)$$

Из (5) вытекают

Утверждение 1. В решетке $R_{k_1,\ldots,k_n}$ $\{\boldsymbol{v} = (v_1, \ldots, v_i, \ldots, v_n), \boldsymbol{v}' = (v_1, \ldots, v_i + 1, \ldots, v_n)\} \in E$ – хорда тогда и только тогда, когда $i = 2, \ldots, n$.

Утверждение 2. В решетке $R_{k_1,\ldots,k_n}$ если $\{\boldsymbol{v}, \boldsymbol{v}'\} \in E$ – хорда длины l, то

$$l \in \left\{ k_1, k_1 \cdot k_2, \ldots, \prod_{i=1}^{n-1} k_i \right\}. \quad (6)$$

Так как

$$\begin{cases} S_{k_1}(R_{k_1,\ldots,k_n}) = k_1 \cdot (k_2 - 1) \cdot k_3 \cdot \ldots \cdot k_n = \prod_{i=1}^{n} k_i \cdot (1 - k_2^{-1}), \\ S_{k_1 \cdot k_2}(R_{k_1,\ldots,k_n}) = k_1 \cdot k_2 \cdot (k_3 - 1) \cdot k_4 \cdot \ldots \cdot k_n = \prod_{i=1}^{n} k_i \cdot (1 - k_3^{-1}), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ S_{\prod_1^{n-1} k_i}(R_{k_1,\ldots,k_n}) = k_1 \cdot k_2 \cdot \ldots \cdot k_{n-1} \cdot (k_n - 1) = \prod_{i=1}^{n} k_i \cdot (1 - k_n^{-1}), \end{cases} \quad (7)$$

то из (1), (6) и (7) вытекает

Теорема 1. Для решетки $R_{k_1,\ldots,k_n}$ истинно равенство

$$S(R_{k_1,\ldots,k_n}) = \left((n-1) - \sum_{i=2}^{n} k_i^{-1} \right) \cdot \prod_{i=1}^{n} k_i \ .$$

Преобразуем решетку $R_{k_1,\ldots,k_n}$ к виду 1-отказоустойчивого графа. Для этого добавим рёбра таким образом, чтобы гамильтонов путь имел стандартный вид: $0 \to 1 \to \ldots \to (n-1)$. Добавим новую вершину n и соединим её с вершинами 0 и $n-1$. Для того чтобы получить минимальный 1-отказоустойчивый граф, необходимо вставить избыточную вершину n в ребро, принадлежащее гамильтонову пути. Осуществим трансляцию дуг по циклической группе автоморфизмов C_n. В результате получим 1-отказоустойчивый граф $R^*_{k_1,\ldots,k_n}$.

Характеристики избыточности при переходе от $R_{k_1,\ldots,k_n}$ к $R^*_{k_1,\ldots,k_n}$ равны

$$S^{\text{раб}}(R^*_{k_1,\ldots,k_n}) = S(R_{k_1,\ldots,k_n}) = \left((n-1) - \sum_{i=2}^{n} k_i^{-1} \right) \cdot \prod_{i=1}^{n} k_i \ ,$$

$$S^{\text{изб}}(R^*_{k_1,\ldots,k_n}) = \prod_{i=1}^{n} k_i \sum_{i=2}^{n} k_i^{-1} + n - 1,$$

$$E^{\text{раб}}(R^*_{k_1,\ldots,k_n}) = (k_1 - 1) \prod_{i=2}^{n} k_i \ ,$$

$$E^{\text{изб}}(R^*_{k_1,\ldots,k_n}) = \prod_{i=2}^{n} k_i \ + 1.$$

Рассмотрим специальные случаи прямоугольной n-мерной решётки $R_{k_1,\ldots,k_n}$, когда $n \in \{2,3\}$.

1. $n = 2$. Решётка R_{k_1,k_2} размера изображена на рис. 1, а граф $R^*_{k_1,k_2}$ – на рис. 2.

Сведём характеристики графа $R^*_{k_1,k_2}$ в табл. 1.

Таблица 1

Характеристики графа $R^*_{k_1,k_2}$

Элементы решётки	Рабочие	Избыточные
Вершины	$k_1 k_2$	1
Рёбра	$(k_1 - 1)k_2$	$k_2 + 1$
Хорды	$(k_2 - 1)k_1$	$k_1 + 1$
Связи (общ.)	$2k_1k_2 - k_1 - k_2$	$k_1 + k_2 + 2$

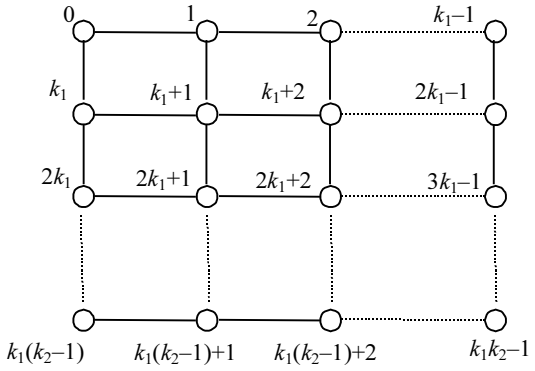


Рис. 1. Решётка R_{k_1,k_2}

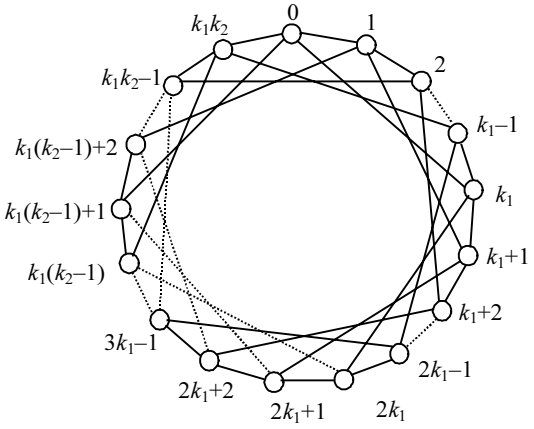


Рис. 2. Граф $R^*_{k_1,k_2}$

Из табл. 1 вытекает, что степени вершинной и рёберной избыточности графа $R^*_{k_1,k_2}$ равны соответственно:

$$\nu_{\text{в}}(R_{k_1,k_2}) = 1 + (k_1k_2)^{-1},$$

$$\nu_{\text{р}}(R_{k_1,k_2}) = 1 + \frac{k_1 + k_2 + 2}{2k_1k_2 - k_1 - k_2}.$$

2. $n = 3$. Решётка R_{k_1,k_2,k_3} изображена на рис. 3. Характеристики графа $R^*_{k_1,k_2,k_3}$ сведены в табл. 2.

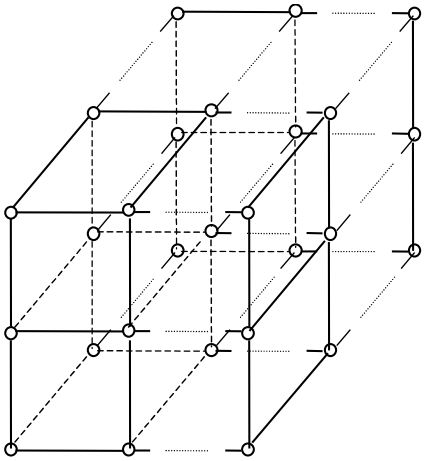


Рис. 3. Решётка R_{k_1,k_2,k_3}

Таблица 2

Характеристики графа $R^*_{k_1,k_2,k_3}$

Элементы решётки	Рабочие	Избыточные
Вершины	$k_1k_2k_3$	1
Рёбра	$(k_1 - 1)k_2k_3$	$k_2k_3 + 1$
Хорды	$2k_1k_2k_3 - k_1(k_2 + k_3)$	$k_1(k_2 + k_3)+2$
Связи (общ.)	$3k_1k_2k_3 - k_1k_2 - k_1k_3 - k_2k_3$	$k_1k_2 + k_1k_3 + k_2k_3+3$

Из табл. 2 вытекает, что степени вершинной и рёберной избыточности равны соответственно:

$$\nu_{\text{в}}(R_{k_1,k_2}) = 1 + (k_1k_2)^{-1},$$

$$\nu_{\text{р}}(R_{k_1,k_2}) = 1 + \frac{k_1k_2 + k_1k_3 + k_2k_3 + 3}{3k_1k_2k_3 - k_1k_2 - k_1k_3 - k_2k_3}.$$

3. Диагональные прямоугольные *n*-мерные решетки

Диагональной прямоугольной *n*-мерной решёткой назовём такой граф $D_{k_1,\ldots,k_n} = (V', E')$, что $V' = \overset{n}{\underset{i=1}{\times}} Z_{k_i}$ и

$$E' = E \cup D^{(1)} \cup D^{(2)},$$

где

$$D^{(1)} = \bigcup_{\substack{i=1,\ldots,n-1 \\ j=i+1,\ldots,n}} D^{(1)}_{ij},\ D^{(2)} = \bigcup_{\substack{i=1,\ldots,n-1 \\ j=i+1,\ldots,n}} D^{(2)}_{ij},$$

а

$$D^{(1)}_{ij} = \{(v_1, \ldots, v_n), (v_1, \ldots, v_i + 1, \ldots, v_j + 1, \ldots, v_n)\},$$

$$D^{(2)}_{ij} = \{(v_1, \ldots, v_i, \ldots, v_j + 1, \ldots, v_n), (v_1, \ldots, v_i + 1, \ldots, v_j, \ldots, v_n)\}.$$

Пусть $\{\boldsymbol{v}, \boldsymbol{v}'\} \in D_{ij}^{(1)}$, где $\boldsymbol{v} = (v_1, \ldots, v_i, \ldots, v_j, \ldots, v_n) \in V$ и $\boldsymbol{v}' = (v_1, \ldots, v_i + 1, \ldots, v_j + 1, \ldots, v_n) \in V$. Из (2) вытекает, что

$$q(\boldsymbol{v}) = v_1 + \ldots + k_1 \cdot \ldots \cdot k_{i-1}\, v_i + \ldots + k_1 \cdot \ldots \cdot k_{j-1} \cdot v_j + \ldots + k_1 \cdot \ldots \cdot k_{(n-1)} \cdot v_n; \quad (8)$$

$$q(\boldsymbol{v}') = v_1 + \ldots + k_1 \cdot \ldots \cdot k_{i-1}\,(v_i + 1) + \ldots + k_1 \cdot \ldots \cdot k_{j-1} \cdot (v_j + 1) + \ldots + k_1 \cdot \ldots \cdot k_{(n-1)} \cdot v_n. \quad (9)$$

Из (8) и (9) вытекает, что

$$| q(\boldsymbol{v}) - q(\boldsymbol{v}') | = (k_1 \cdot \ldots \cdot k_{i-1})(1 + k_i \cdot \ldots \cdot k_{j-1}) \ \ (i,j = 2, \ldots, n).$$

Число таких диагоналей равно

$$S_{D_{ij}^{(1)}} = \prod_{h=1}^{n} k_h \cdot \sum_{i<j} ((1 - k_j^{-1})(1 - k_i^{-1})).$$

Пусть $\{\boldsymbol{v}, \boldsymbol{v}'\} \in D_{ij}^{(2)}$, где $\boldsymbol{v} = (v_1, \ldots, v_i, \ldots, v_j + 1, \ldots, v_n) \in V$ и $\boldsymbol{v}' = (v_1, \ldots, v_i + 1, \ldots, v_j, \ldots, v_n) \in V$. Из (2) вытекает, что

$$q(\boldsymbol{v}) = v_1 + k_1 \cdot \ldots \cdot k_{i-1}\, v_i + \ldots + k_1 \cdot \ldots \cdot k_{j-1} \cdot (v_j + 1) + \ldots + k_1 \cdot \ldots \cdot k_{(n-1)} \cdot v_n; \quad (10)$$

$$q(\boldsymbol{v}') = v_1 + k_1 \cdot \ldots \cdot k_{i-1}\,(v_i + 1) + \ldots + k_1 \cdot \ldots \cdot k_{j-1} \cdot v_j + \ldots + k_1 \cdot \ldots \cdot k_{(n-1)} \cdot v_n. \quad (11)$$

Из (10) и (11) вытекает, что

$$| q(\boldsymbol{v}) - q(\boldsymbol{v}') | = (k_1 \cdot \ldots \cdot k_{i-1})(1 - k_i \cdot \ldots \cdot k_{j-1}) \ \ (i,j = 2, \ldots, n).$$

Нетрудно убедиться в том, что

$$S_{D_{ij}^{(1)}} = S_{D_{ij}^{(2)}} = \prod_{h=1}^{n} k_h \cdot \sum_{i<j} (1 - k_j^{-1})(1 - k_i^{-1}),$$

где

$$S_{D_{ij}^{(n)}} = \left| D_{ij}^{(n)} \right| \ (n = 1, 2).$$

После трансляции дуг по циклической группе автоморфизмов C_n получим 1-отказоустойчивый граф $D^*_{k_1,\ldots,k_n}$. Характеристики избыточности при переходе от решетки $D_{k_1,\ldots,k_n}$ к графу $D^*_{k_1,\ldots,k_n}$ равны:

$$E^{\text{раб}}(D^*_{k_1,\ldots,k_n}) = (k_1 - 1) \prod_{i=2}^{n} k_i,$$

$$E^{\text{изб}}(D^*_{k_1,\ldots,k_n}) = \prod_{i=2}^{n} k_i + 1,$$

$$S^{\text{раб}}(D^*_{k_1,\ldots,k_n}) = \left(n - 1 - \sum_{i=2}^{n} k_i^{-1} + 2 \sum_{i<j} (1 - k_j^{-1})(1 - k_i^{-1}) \right) \prod_{i=1}^{n} k_i,$$

$$S^{\text{изб}}(D^*_{k_1,\ldots,k_n}) = \left(2n + \sum_{i=2}^{n} k_i^{-1} - 2 \sum_{i<j} (1 - k_j^{-1})(1 - k_i^{-1}) \right) \prod_{h=1}^{n} k_h + 3n - 3.$$

Рассмотрим специальные случаи диагональной прямоугольной n-мерной решётки $D_{k_1,\ldots,k_n}$, когда $n \in \{2, 3\}$.

1. $n = 2$. Решётка D_{k_1,k_2} изображена на рис. 4, а граф $D^*_{k_1,k_2}$ – на рис. 5.

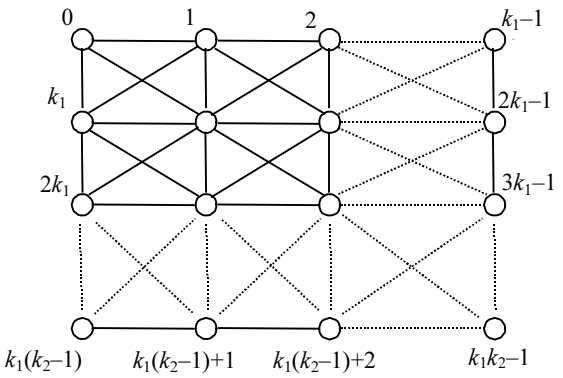


Рис. 4. Решётка D_{k_1,k_2}

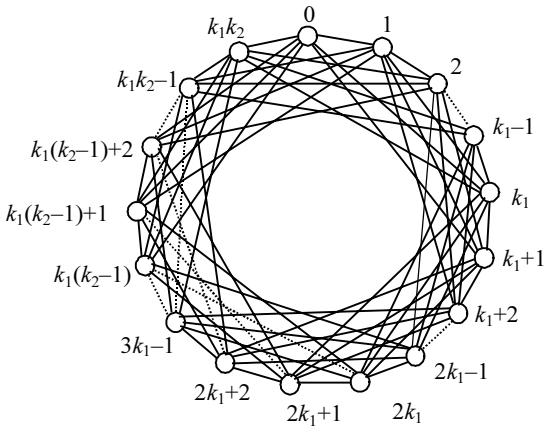


Рис. 5. Граф $D^*_{k_1,k_2}$

Характеристики графа $D^*_{k_1,k_2}$ сведём в табл. 3.

Таблица 3

Характеристики графа $D^*_{k_1,k_2}$

Элементы решётки	Рабочие	Избыточные
Вершины	k_1k_2	1
Рёбра	$(k_1-1)k_2$	k_2+1
Хорды	$3k_1k_2-3k_1-2k_2+2$	$3k_1+2k_2+1$
Связи (общ.)	$4k_1k_2-3(k_1+k_2)+2$	$3k_1+3k_2+2$

Из табл. 3 вытекает, что степени вершинной и рёберной избыточности графа $D^*_{k_1,k_2}$ равны соответственно:

$$\nu_{\text{в}}(D^*_{k_1,k_2}) = 1+(k_1k_2)^{-1},$$

$$\nu_{\text{р}}(D^*_{k_1,k_2}) = 1+\frac{3k_1+3k_2+2}{4k_1k_2-3k_1-3k_2+2}.$$

2. $n = 3$. Характеристики графа $D^*_{k_1,k_2,k_3}$ сведены в табл. 4.

Таблица 4

Характеристики графа $R^*_{k_1,k_2,k_3}$

Элементы решётки	Рабочие	Избыточные
Вершины	$k_1k_2k_3$	1
Рёбра	$(k_1-1)k_2k_3$	k_2k_3+1
Хорды	$8k_1k_2k_3-5k_1k_3-5k_1k_2-4k_2k_3+2(k_1+k_2+k_3)$	$5k_1(k_2+k_3)+4k_2k_3-2(k_1+k_2+k_3)-2k_1k_2k_3+6$
Связи (общ.)	$9k_1k_2k_3-5k_1k_3-5k_1k_2-5k_2k_3+2(k_1+k_2+k_3)$	$5k_1(k_2+k_3)+5k_2k_3-2(k_1+k_2+k_3)-2k_1k_2k_3+7$

Из табл. 4 вытекает, что степени вершинной и рёберной избыточности равны соответственно:

$$\nu_{\text{в}}(D^*_{k_1,k_2,k_3}) = 1+(k_1k_2)^{-1},$$

$$\nu_{\text{р}}(D^*_{k_1,k_2,k_3}) = 1+\frac{5(k_1k_2+k_1k_3+k_2k_3)-2(k_1+k_2+k_3)-2k_1k_2k_3+7}{9k_1k_2k_3-5(k_1k_3+k_1k_2+k_2k_3)+2(k_1+k_2+k_3)}.$$

Заключение

В работе исследованы характеристики избыточности при преобразовании простых прямоугольных n-мерных решёток и диагональных прямоугольных n-мерных решёток в 1-отказоустойчивые минимальные графы. Полученные оценки являются верхними границами избыточности при преобразовании в 1-отказоустойчивый граф любого графа, вложимого в рассмотренные решётки. Одно из направлений дальнейших исследований состоит в оценке избыточности при преобразовании более сложных регулярных классических структур в 1-отказоустойчивые графы. Второе направление исследований состоит в изучении характеристик избыточности при преобразовании регулярных структур в m-отказоустойчивые графы, где $m \geq 2$.

ЛИТЕРАТУРА

1. *Каравай М.Ф.* Общий подход к построению отказоустойчивых цифровых систем // Вестник ТГУ. Приложение. 2004. № 9. С. 123 – 134.
2. *Каравай М.Ф.* Инвариантно-групповой подход к исследованию k-отказоустойчивых структур // Автоматика и телемеханика. 2000. № 1. С. 144 – 156.
3. *Каравай М.Ф.* Применение теории симметрии к анализу и синтезу отказоустойчивых систем // Автоматика и телемеханика. 1996. № 6. С. 159 – 173.
4. *Bollobas B.* Modern graph theory. N.Y.: Springer Verlag, 1998. 394 p.

2008 Прикладная теория графов № 1(1)
DOI 10.17223/20710410/1/18
УДК 519.17

МИНИМАЛЬНЫЕ ПРИМИТИВНЫЕ РАСШИРЕНИЯ ОРИЕНТИРОВАННЫХ ГРАФОВ

В.Н. Салий

Саратовский государственный университет им. Н.Г.Чернышевского

E-mail: SaliiVN@info.sgu.ru

Получено решение задачи о построении минимальных примитивных расширений для некоторых типов ациклических графов (исходящие деревья, линейные и многоугольные графы).

Ключевые слова: *примитивный граф*, *минимальное примитивное расширение*, *дерево*, *линейный граф*, *многоугольный граф.*

Под ориентированным графом (далее – граф) понимается пара $G = (V, \alpha)$, где V – конечное непустое множество и $\alpha \subseteq V^2$ – отношение на нем. Элементы множества V называются вершинами графа, а пары, входящие в отношение смежности α, – его дугами. Если $(u, v) \in \alpha$, то говорят, что вершина u является началом дуги (u, v), а вершина v – ее концом. При $u = v$ получается петля (u, u). Считаем, что каждая вершина графа инцидентна некоторой дуге, т.е. является началом или концом некоторой дуги.

Вершина v по определению достижима из вершины u за $k \geq 1$ шагов, если существует последовательность примыкающих дуг (маршрут) $(w_0, w_1), (w_1, w_2), \ldots, (w_{k-1}, w_k)$ (в краткой записи $w_0w_1w_2\ldots w_{k-1}w_k$), где $w_0 = u$ и $w_k = v$. Если A – матрица смежности графа G, т.е. двоичная булева матрица, представляющая отношение α, то последнее определение означает, что на пересечении строки, соответствующей элементу u, и столбца, соответствующего элементу v, в матрице-степени A^k стоит 1.

Для дальнейшего выделим два типа маршрутов с неповторяющимися вершинами. Это n-элементная цепь $P_n = v_1v_2\ldots v_n$ и n-элементный контур $C_n = v_1v_2\ldots v_nv_1$ (начало и конец совпадают).

Граф называется примитивным, если существует целое число $r \geq 1$ такое, что каждая вершина графа достижима из любой вершины за r шагов (иначе говоря, если в матрице A^r все элементы равны 1). Таким образом, каждый примитивный граф является сильно связным (любые две вершины взаимно достижимы), но обратное, конечно, неверно: например, никакой контур не является примитивным графом.

Пусть K – некоторый класс графов и $G = (V, \alpha)$ – произвольный граф. Граф $G' = (V, \alpha')$ с тем же множеством вершин называется K-расширением графа G, если $\alpha \subseteq \alpha'$ и $G' \in K$. Задача о минимальных K-расширениях данного графа G состоит в следующем: как добавить к G наименьшее возможное число дуг, чтобы получился K-граф? Можно интересоваться, например, минимальными сильно связными, или эйлеровыми, или гамильтоновыми и т.п. расширениями заданного графа. В [1] эта задача была решена для класса K идемпотентных графов, т.е. графов с идемпотентной ($A^2 = A$) матрицей смежности: предложена процедура построения минимального идемпотентного расширения графа.

В настоящей заметке в качестве целевого класса K выбран класс примитивных графов. Основным инструментом в доказательствах является следующий критерий примитивности (см., например, [2]): сильно связный граф примитивен тогда и только тогда, когда наибольший общий делитель длин всех его контуров равен 1. В частности, если в сильно связном графе имеется хотя бы одна петля, граф будет примитивным. Отбрасывая этот тривиальный случай, будем рассматривать только беспетельные графы. Далее, ни один из двухвершинных беспетельных графов (цепь P_2 и контур C_2) не имеет примитивных расширений, так что все рассматриваемые графы будут иметь число вершин $n \geq 3$.

В силу приведенного выше критерия, приступая к отысканию минимальных примитивных расширений графа, представляется естественным сначала посмотреть, нет ли примитивных графов среди его минимальных сильно связных расширений, и уже затем, если таковых не найдется, продолжить процесс добавления дуг. Оказывается, достаточно будет присоединить всего лишь одну дугу.

Теорема 1. Если сильно связный граф не является примитивным, то его минимальное примитивное расширение получается добавлением одной дуги.

Доказательство. Пусть G – сильно связный не примитивный граф. Вследствие сильной связности каждая его дуга содержится в некотором контуре (см. [3, теорема 8.1.5]). Предположим, что в G имеется контур $C_k = v_1v_2\ldots v_kv_1$ длины $k \geq 3$. Присоединив к графу G дугу v_1v_3, обнаруживаем контур $v_1v_3\ldots v_kv_1$, длина которого $k - 1$ взаимно проста с k, – полученный граф примитивен.

Допустим теперь, что в графе G все контуры имеют длину 2 и $C_2 = v_1v_2v_1$ – один из них. Так как G имеет не менее трех вершин, в нем существует вершина v_3, смежная с v_1 или с v_2. Пусть дугой в G будет v_2v_3. Вер-

шина v_2 достижима из v_3. Значит, в G есть контур, содержащий дугу v_2v_3. По условию, им может быть только контур $v_2v_3v_2$. Присоединив к графу G дугу v_3v_1, получаем контур $v_1v_2v_3v_1$ длины 3 – полученный граф примитивен. ■

Пусть $\mu(G)$ обозначает количество добавочных дуг в минимальном сильно связном расширении $G' = (V, \alpha')$ графа $G = (V, \alpha)$, т.е. $\mu = |\alpha'| - |\alpha|$, и пусть аналогично $\nu(G)$ – количество добавочных дуг в минимальном примитивном расширении этого графа. Из теоремы 1 вытекает

Следствие. Для любого графа G имеет место неравенство $\nu \le \mu + 1$. ■

Турниром называется беспетельный граф, в котором любые две различные вершины соединены в точности одной дугой. Единственный двухвершинный турнир – это цепь P_2, и она не имеет примитивных расширений. К числу сильно связных турниров относится контур $C_3 = v_1v_2v_3v_1$. Сам он не является примитивным графом, и, присоединив к нему, как в доказательстве теоремы 1, дугу v_1v_3, получим его минимальное примитивное расширение, т.е. в этом случае $\nu = \mu + 1$.

Теорема 2. Каждый сильно связный турнир с $n \ge 4$ вершинами примитивен. Если T – не сильно связный турнир с $n \ge 3$ вершинами, то для него $\nu = \mu + 1$.

Доказательство. Пусть T – сильно связный турнир с $n \ge 4$ вершинами. По теореме Мозера (см. [4, теорема 3.43]) в нем имеются контуры всех длин $3 \le k \le n$, откуда и следует его примитивность. Пусть T – не сильно связный турнир с $n \ge 3$ вершинами. По теореме Редеи (см. [4, теорема 3.42]) в нем существует гамильтонова (т.е. проходящая через все вершины) цепь $P = v_1v_2\ldots v_n$. Добавим к ней дугу v_nv_1. Полученный граф T' является минимальным сильно связным расширением турнира T. Будет ли он примитивным? Если из T удалить дугу v_1v_n (она есть в T), то получится сильно связный и, значит, примитивный турнир. Возвращаясь к T' восстановлением дуги v_1v_n, мы, конечно, не утратим свойство примитивности. ■

Далее будем рассматривать только бесконтурные графы.

Вершина графа называется источником, если в нее не входит ни одна дуга, и называется стоком, если из нее не исходит ни одна дуга. Как известно, в бесконтурном графе любая вершина достижима из подходящего источника и из любой вершины достижим подходящий сток. Обозначив через l количество источников, а через m – количество стоков в графе G, получаем, что $\mu(G) \ge \max(l, m)$ (в минимальном сильно связном расширении графа G в каждый G-источник должна войти дуга и из каждого G-стока должна выйти дуга).

Теорема 3. Для цепи P_n, $n \ge 3$, имеем $\nu = \mu + 1 = 2$.

Доказательство. Присоединяя к цепи $P_n = v_1v_2\ldots v_n$, $n \ge 3$, дугу v_nv_1, получаем контур C_n. Далее, как в доказательстве теоремы 1, добавляем к C_n дугу v_1v_3. ■

Граф Tr называется выходящим деревом, если в нем существует точно один источник, а в любую другую вершину входит точно одна дуга. Стоки выходящего дерева называются его листьями.

Следующее предложение показывает, что среди минимальных сильно связных расширений выходящего дерева обязательно найдутся примитивные.

Теорема 4. Для выходящего дерева с $m \ge 2$ листьями $\nu = \mu = m$.

Доказательство. Пусть Tr_m – выходящее дерево с корнем v_0 и листьями $v_1, v_2,\ldots, v_m$. Пусть h_1 – высота листа v_1, т.е. длина цепи, соединяющей v_0 с v_1, а h_m – высота листа v_m. Среди $h_1 + 1$ последовательных чисел $(h_m + 1) + 1, (h_m + 1) + 2, \ldots, (h_m + 1) + (h_1 + 1)$ выберем наименьшее, взаимно простое с $h_1 + 1$. Пусть это будет $(h_m + 1) + i$ (заметим, что $i < h_1 + 1$). Проведем дугу из каждого листа $v_1, v_2, \ldots, v_{m-1}$ в корень v_0 и из листа v_m в вершину с номером $(h_1 + 1) - i$, входящую в состав цепи $u_0u_1\ldots u_{h1-1}u_{h1}$, соединяющей v_0 с v_1 (в ней $u_0 = v_0$ и $u_{h_1} = v_1$). Получается сильно связный граф с числом добавочных дуг, равным числу стоков (листьев), он будет минимальным сильно связным расширением для Tr_m. При этом в полученном графе есть контур $u_0u_1\ldots u_{h1}u_0$ длины h_1+1 и есть контур длины $(h_m+1) + i$, составленный из цепи, соединяющей v_0 с v_m, дуги из v_m в $u_{(h_1+1)-i}$, цепи из $u_{(h_1+1)-i}$ в v_1 и дуги v_1v_0. ■

Линейный граф с n вершинами – это граф, полученный из цепи P_n переориентацией некоторых ее дуг. Если под степенью вершины понимать количество дуг, которым она инцидентна, то в линейном графе каждая вершина, кроме двух, называемых крайними, имеет степень 2, а крайние вершины имеют степень 1.

Линейный граф по определению относится к типу I, если его крайние вершины обе являются источниками или обе являются стоками, и относится к типу II, если одна из его крайних вершин – источник, а другая – сток. Для линейных графов типа I решение задачи о минимальных примитивных расширениях дает

Теорема 5. Для линейного графа типа I $\nu = \mu = \max(l, m)$, где l и m – соответственно количество источников и стоков.

Доказательство. Пусть $u_1P_1v_1Q_1u_2P_2v_2\ldots Q_ku_{k+1}$ – линейный граф типа I с источниками u_i, $1 \le i \le k + 1$, стоками v_j, $1 \le j \le k$, и цепями P_j, Q_j, ведущими из источников в стоки. Сильно связное расширение данного графа имеет не менее чем $l = k + 1$ добавочных дуг. Покажем, что среди его минимальных сильно связных расширений имеются примитивные. Проведем дугу из каждого стока v_i в источник u_i, $1 \le i \le k$. Пусть в по-

лученном графе G длина контура $C = u_1P_1v_1u_1$ равна s. Положим $t = k + \Sigma^k_{i=1}|Q_i|$, где $|Q_i|$ – длина цепи Q_i, $1 \le i \le k$. Пусть $d = $ НОД(s, t).

1) $d = 1$. К графу G присоединим дугу v_1u_{k+1}. В полученном сильно связном графе $G+v_1u_{k+1}$ есть контур C длины s и контур $C' = v_1u_{k+1}Q_kv_ku_k\ldots Q_1v_1$ длины t, откуда следует примитивность этого графа. Итак, в этом случае $\nu = \mu = k + 1$.

2) $d \ne 1$. Запишем цепь P_1 в виде $P_1 = u'_1u'_2\ldots u'_{p-1}u'_p$, где $u'_1 = u_1$, $u'_p = v_1$. Среди $p - 1$ последовательных чисел $t + 1$, $t + 2$, …, $t + (p - 1)$ выберем наименьшее взаимно простое с p. Пусть это будет, например, $t + i$. К графу G присоединим дугу u'_iu_{k+1}. В полученном сильно связном графе есть контур C длины s и контур $C_i = u'_iu_{k+1}Q_kv_ku_k\ldots Q_1v_1Cu'_i$ длины $t + i$, откуда следует примитивность этого графа. И в этом случае $\nu = \mu = k + 1$.

Аналогично добавлением $k + 1$ дуг получается примитивное расширение для линейного графа типа I, крайние вершины которого являются стоками. ■

Для линейных графов типа II полное решение задачи о минимальных примитивных расширениях пока не получено.

Пусть P – линейный граф типа II и d обозначает наибольший общий делитель всех увеличенных на единицу длин его цепей.

Теорема 6. Если у линейного графа P типа II $d \ne 1$, то для этого графа $\nu = \mu + 1 = l + 1$, где l – количество источников (и стоков) в P.

Доказательство. Пусть $P = u_1P_1v_1Q_1u_2P_2v_2\ldots P_lv_l$. Так как P имеет l источников и l стоков, то $\mu \ge l$. На самом деле, у графа P существует минимальное сильно связное расширение с l добавочными дугами: оно получается присоединением к P всех дуг вида v_iu_j, где $i + j = l + 1$. Пусть P' – некоторое минимальное сильно связное расширение графа P. Рассмотрим в нем произвольный контур C. Так как в C из каждого стока графа P идет дуга в подходящий источник, то $C = v'_1u'_1R_1v'_2u'_2R_2\ldots u'_kR_kv'_1$, где R_i, $1 \le i \le k$, – цепи из P. Длина контура C равна $k + \Sigma^k_{i=1}|R_i| = \Sigma^k_{i=1}(1 + |R_i|)$ и, следовательно, делится на d. Таким образом, минимальное сильно связное расширение P' графа P не будет примитивным.

Согласно теореме 1, существует минимальное примитивное расширение графа P с $l + 1$ добавочными дугами. Построим его следующим образом. Из каждого стока v_i графа P проведем дугу в источник u_{i+1}, $1 \le i \le l - 1$, и проведем две дуги из v_l: в источник u_1 и в вершину u' – следующую за u_1 в цепи P_1 (возможно, окажется, что $u' = v_1$). В полученном сильно связном графе P' имеем контур $C = u_1P_1v_1u_2P_2\ldots P_lv_lu_1$ длины $l + \Sigma^l_{i=1}|P_i|$ и контур $C' = u'Cv_lu'$ длины на единицу меньше. Значит, P' – примитивный граф.

Аналогично добавлением $l + 1$ дуг получается примитивное расширение для линейного графа типа II $v_1Q_1u_1P_1v_2Q_2u_2\ldots Q_lu_l$, в котором $d \ne 1$. ■

Среди линейных графов типа II с $d = 1$ существуют такие, у которых есть минимальные сильно связные расширения, являющиеся примитивными, и существуют такие, у которых все минимальные сильно связные расширения не примитивны. Два простейших примера: у графа $P_1 = u_1 \to v_1 \leftarrow u_2 \to w \to v_2$ единственное минимальное сильно связное расширение (оно получается добавлением дуг v_1u_2 и v_2u_1) примитивно, а у графа $P_2 = u_1 \to v_1 \leftarrow u_2 \to v_2$ единственное минимальное сильно связное расширение (и оно получается добавлением одноименных дуг) не примитивно.

Многоугольный граф с $n \ge 3$ вершинами – это граф, полученный из контура C_n переориентацией некоторых его дуг. В многоугольном графе каждая вершина имеет степень 2. Количество источников в многоугольном графе равно количеству стоков. Пусть φ – некоторая биекция между множеством стоков и множеством источников данного многоугольного графа C. Если к C присоединить все дуги вида $v\varphi(v)$, где v – сток, получится расширение графа C, назовем его φ-расширением. В отличие от линейных графов типа II (у них тоже одинаковое число источников и стоков), имеет место следующий факт.

Теорема 7. Для любой биекции φ между множеством стоков и множеством источников многоугольного графа соответствующее его φ-расширение является минимальным сильно связным расширением.

Доказательство. Пусть $C = u_1P_1v_1Q_1u_2P_2\ldots v_lQ_lu_l$ – многоугольный граф, где u_i, v_i, $1 \le i \le l$, соответственно источники и стоки, P_i, Q_i, $1 \le i \le l$ – цепи, ведущие из источников в стоки. Пусть, далее, φ – произвольная биекция между множеством стоков и множеством источников графа C. Покажем, что, добавив к C всевозможные дуги вида $v\varphi(v)$, где v – сток в C, получим сильно связный граф C'. Он и будет минимальным сильно связным расширением, так как количество добавляемых к C дуг равно l – количеству источников (и стоков) в C.

Для нашей цели достаточно установить, что в C' из каждого стока v графа C достижим любой источник этого графа.

Пусть v'_1 – произвольный сток в C. Будем строить цепь $R_1 = v'_1\varphi(v'_1)P'_1v'_2\varphi(v'_2)P'_2\ldots$, где P'_j – соответствующие цепи P_i в графе C. Цепь R_1 закончится фрагментом $P'_{l-1}v'_l\varphi(v'_l)$, где источник $\varphi(v'_l)$ таков, что цепь

P'_l имеет своим концом один из стоков, уже встретившихся в R_1. Обозначим через V_1 множество источников и стоков графа C, входящих в состав R_1. Ясно, что все источники из V_1 достижимы в C' из стока v'_1.

Предположим, что некоторый источник u не входит в V_1. Если все цепи графа C, исходящие из источников, отмеченных в составе V_1, имеют своими концами стоки из V_1, то, имея $2l$ таких цепей и l стоков в V_1, получаем, что в каждый такой сток входят точно две цепи из указанных источников. Это означает, что множество вершин V_1 и его дополнение (в C) не соединяет ни одна дуга графа C. Другими словами, C оказывается несвязным графом, что невозможно. Значит, в V_1 существует по крайней мере один источник, из которого в графе C исходит цепь Q'_1, имеющая концом некоторый сток v''_1, не принадлежащий V_1. Но тогда и источник $\varphi(v''_1)$ не входит в V_1. По образцу цепи R_1 строим цепь $R_2 = v''_1\varphi(v''_1)P''_1v''_2\varphi(v''_2)P''_2\ldots$ Пусть множество вершин V_2 состоит из источников и стоков графа C, входящих в цепи R_1 и R_2. Очевидно, что все источники, попавшие в V_2, достижимы в C' из стока v'_1. Если $u \notin V_2$, продолжим процесс и построим цепь R_3 и множество вершин V_3 и т.д. При этом количество использованных источников на каждом шаге увеличивается по крайней мере на единицу. Рано или поздно в их числе окажется и данный источник u.

Таким образом, в графе C' из любого стока графа C можно пройти по дугам в любой источник графа C. Так как C – бесконтурный граф, то в нем из каждой вершины достижим некоторый сток и каждая вершина достижима из подходящего источника. Присоединяя к C все дуги вида $v\varphi(v)$ и ссылаясь на доказанное свойство, получаем сильную связность графа C'. ■

Для многоугольных графов полное решение задачи о минимальных примитивных расширениях пока не получено.

Пусть d обозначает наибольший общий делитель всех увеличенных на единицу длин цепей многоугольного графа C.

Теорема 8. Если у многоугольного графа $d \neq 1$, то для него $\nu = \mu + 1$.

Доказательство. Пусть C' – произвольное минимальное сильно связное расширение многоугольного графа $C = u_1P_1v_1Q_1u_2P_2\ldots v_lQ_lu_l$, имеющего $d \neq 1$. Граф C' получается из C добавлением $\mu = l$ дуг. Любой контур в C' имеет вид $u'_1R_1v'_1\,u'_2\,R_2\ldots R_kv'_ku'_1$, где u'_i, v'_i, $1 \le i \le k$, – источники и стоки в C, а R_i – какие-то цепи графа C. Длина этого контура равна $k + \Sigma^k_{i=1}|R_i| = \Sigma^k_{i=1}(|R_i| + 1)$. Это число делится на d. Следовательно, никакое минимальное сильно связное расширение графа C не может быть примитивным. Но тогда, согласно теореме 1, существует его примитивное расширение, имеющее $\nu = l + 1 = \mu + 1$. ■

Среди многоугольных графов с $d = 1$ существуют такие, у которых есть минимальные сильно связные расширения, являющиеся примитивными, и существуют такие, у которых все минимальные сильно связные расширения не примитивны. Примером многоугольного графа первого вида служит граф $C = u_1P_1v_1Q_1u_2P_2v_2Q_2u_1$, где $|P_1| = |P_2| = 1$, $|Q_1| = |Q_2| = 2$. Если к графу C присоединить дуги v_1u_2 и v_2u_1, получается его минимальное сильно связное расширение, которое оказывается примитивным (есть контуры длин 3 и 4). Примером многоугольного графа второго вида служит граф того же вида, где $|P_1| = 1$, $|Q_1| = |Q_2| = 2$, $|P_3| = 3$. У него два минимальных сильно связных расширения, и оба они не примитивны: если к C добавить дуги v_1u_1 и v_2u_2, в полученном графе C' контуры имеют длины 2, 4 и 6, а если к C добавить дуги v_1u_2 и v_2v_1, в полученном графе C' контуры имеют длины 3, 3 и 6.

ЛИТЕРАТУРА

1. *Салий В.Н.* Функциональная отказоустойчивость и оптимальные реконструкции графовых систем по заданным параметрам // Вестник ТГУ. Приложение. 2007. № 23. С. 253 – 256.
2. *Beasley Le Roy B.*, *Kirkland S.* A note on *k*-primitive directed graphs // Linear Algebra and Appl. 2003. V. 373. P. 67 – 74.
3. *Оре О.* Теория графов. М.: Наука, 1968.
4. *Богомолов А.М.*, *Салий В.Н.* Алгебраические основы теории дискретных систем. М.: Наука, 1997.

ПРИКЛАДНАЯ ТЕОРИЯ АВТОМАТОВ

DOI 10.17223/20710410/1/19

УДК 519.7

К ОПИСАНИЮ ПРОГРЕССИВНЫХ РЕШЕНИЙ ПАРАЛЛЕЛЬНОГО АВТОМАТНОГО УРАВНЕНИЯ[1]

В.Г. Бушков

Томский государственный университет

E-mail: bushkov@sibmail.com

Работа посвящена нахождению наибольшего прогрессивного решения параллельного автоматного уравнения $C \lozenge X \cong S$, где C и S – конечные полностью определенные автоматы, на основе удаления непрогрессивных последовательностей из наибольшего решения уравнения. Именно прогрессивные решения, в композиции которых с автоматом C отсутствуют заведомо тупиковые ситуации, интересны с практической точки зрения.

***Ключевые слова*:** *конечный автомат*, *параллельная композиция*, *автоматное уравнение*, *прогрессивное решение.*

Многие задачи синтеза и анализа систем логического управления сводятся к решению автоматного уравнения $C \lozenge X \cong S$, где C и S – конечные автоматы, $\lozenge$ – операция параллельной композиции, которая соответствует поочередной работе автоматов-компонент [1]. Известно, что разрешимое уравнение имеет наибольшее решение, которое можно рассматривать как резервуар для выбора наилучшего в некотором смысле решения. Однако не каждое решение интересно с практической точки зрения. Особый интерес представляют так называемые прогрессивные решения, т.е. решения, в композиции которых с автоматом C отсутствуют тупики и осцилляции. В работе [2] наибольшее прогрессивное решение строится для случая, когда компоненты уравнения являются полуавтоматами, и предлагается алгоритм нахождения такого решения на основе расщепления состояний наибольшего решения. В данной работе мы предлагаем другой подход к построению наибольшего прогрессивного решения, который основан на удалении из наибольшего решения так называемых не прогрессивных входо-выходных последовательностей. Поскольку в общем случае число последовательностей в наибольшем решении бесконечно, то нетривиальным является вопрос о сходимости предлагаемого алгоритма, и мы показываем, что предложенный алгоритм сходится.

1. Основные определения

Алфавитом A называется непустое конечное множество символов. Обозначим через A^* множество всех конечных слов над алфавитом A. Подмножество $L \subseteq A^*$ называется *языком*. Рассмотрим непустое подмножество A_1 множества A и отображение h: $A \to A_1 \cup \{\varepsilon\}$, где ε – пустое слово. Мы полагаем, что $h(\alpha) = \alpha$ для всех $\alpha \in A_1$ и $h(\alpha) = \varepsilon$ для всех $\alpha \in A \backslash A_1$, причём h расширяется на слова по правилу: $h(\alpha\beta) = h(\alpha)h(\beta)$ и $h(\varepsilon) = \varepsilon$. Тогда язык $L_{\Downarrow A_1} = \{h(\beta) : \beta \in L\}$ называется *ограничением* языка L на алфавит A_1. Пусть теперь язык L определен над алфавитом $A_2 = A \backslash A_1$. Рассмотрим отображение ψ: $A_2 \to 2^{A^*}$, такое, что $\psi(\alpha) = \{\gamma\alpha\beta$: $\gamma, \beta \in A_1^*\}$. Тогда язык $L_{\Uparrow A} = \{\psi(\beta)$: $\beta \in L\}$ есть *распространение* языка L на алфавит A. По определению, распространение пустого языка есть пустой язык. Пусть язык L_1 определен над алфавитом A_1, язык L_2 определен над алфавитом A_2, $A = A_1 \cup A_2$ и S – непустое подмножество множества A. Параллельной композицией $L_1 \lozenge_S L_2$ называется язык $(L_{1\Uparrow A} \cap L_{2\Uparrow A})_{\Downarrow S}$. Для языка L над алфавитом A через $Init(L) = \{\alpha \in A^* \mid \exists \beta \in A^*, \alpha\beta \in L\}$ обозначим язык, состоящий из всех префиксов всех слов языка L.

Автоматом называется пятёрка $\mathcal{S} = (S, I, O, T_S, s_0)$, где S – конечное множество состояний, I – конечный входной алфавит, O – конечный выходной алфавит, $T \subseteq I \times S \times S \times O$ – отношение переходов, s_0 – начальное состояние. Автомат называется *полностью определенным*, если $\forall (i,s) \in I \times S \;\; \exists (s',o) \in S \times O \;\; ((i,s,s',o) \in T)$; иначе автомат называется *частичным*. Рассмотрим автоматы $\mathcal{S} = (S, I, O, T_S, s_0)$ и $\mathcal{Q} = (Q, X, Y, T_Q, q_0)$. Если

[1] Работа частично поддержана грантом РФФИ 06-08-89500.

множества $I \cap X$ и $O \cap Y$ не являются пустыми, то пересечением автоматов $S \cap Q$ называется наибольший связный подавтомат автомата $(S \times Q, I \cap X, O \cap Y, H, s_0q_0)$, в котором отношение H определено следующим образом: $(i, sq, s'q', o) \in H \Leftrightarrow (i, s, s', o) \in T_S$ & $(i, q, q', o) \in T_Q$.

*Полуавтомат*ом называется пятёрка $M = (M, I, T_M, m_0, Q)$, где M – конечное множество состояний, I – конечный алфавит, $T \subseteq I \times S \times S$ – отношение переходов, m_0 – начальное состояние, $Q \subseteq M$ – подмножество финальных состояний. Языком полуавтомата $L \subseteq I^*$ называется множество всех последовательностей, которые переводят полуавтомат из начального состояния в одно из финальных состояний. Любой автомат $S = (S, I, O, T_S, s_0)$ можно представить полуавтоматом M_S, «растягивая» каждый переход в автомате на два перехода, один из которых помечен входным, а другой – выходным символом. Формально, $M_S = (S \cup S \times I, I \cup O, \Delta, s_0, S)$, где $(i,s,(s,i)) \in \Delta \wedge (o,(s,i),s') \in \Delta \Leftrightarrow (i,s,s',o) \in T_S$. Финальными объявляются состояния из множества S. *Языком* автомата S называется язык соответствующего полуавтомата M, т.е. языком автомата S называется множество входо-выходных последовательностей, допустимых в начальном состоянии автомата. Далее язык полуавтомата M обозначается M. Полуавтомат $Init(M)$ с языком $Init(M)$ получается из полуавтомата M удалением всех состояний, в которых генерируется пустой язык, и объявлением всех оставшихся состояний финальными. Полностью определенные автоматы A и B называются *эквивалентными* ($A \cong B$), если их языки совпадают, т.е. $A \cong B \Leftrightarrow L(A) = L(B)$. Автомат B называется *редукцией* автомата A ($B \leq A$), если язык автомата B содержится в языке автомата A, т.е. $B \leq A \Leftrightarrow L(B) \subseteq L(A)$. Если из контекста понятно, является рассматриваемый объект автоматом или полуавтоматом, то для простоты изложения автомат и соответствующий ему полуавтомат обозначаются одним символом.

Пусть I_1, I_2, V, U, O_1, O_2 – попарно непересекающиеся конечные множества, некоторые из которых могут быть пустыми. *Параллельной композицией* автоматов A с входным алфавитом $I_1 \cup V$ и выходным алфавитом $O_1 \cup U$ и B с входным алфавитом $I_2 \cup U$ и выходным алфавитом $O_2 \cup V$ называется автомат $A \lozenge_{Ext} B$, далее просто $A \lozenge B$, язык которого есть $(L(A) \lozenge_{Ext} L(B)) \cap (IO)^*$, далее просто $(L(A) \lozenge L(B)) \cap (IO)^*$, где $I = I_1 \cup I_2$, $O = O_1 \cup O_2$, $Ext = I \cup O$[1]. *Автоматным уравнением* называется выражение вида $C \lozenge X \cong S$, где $C = (C, I_1 \cup V, O_1 \cup U, T_C, c_0)$, $S = (S, I_1 \cup I_2, O_1 \cup O_2, T_S, s_0)$ – конечные автоматы, которые соответственно называются *контекстом* и *спецификацией* [2], $X = (X, I_2 \cup U, O_2 \cup V, T_X, x_0)$ – неизвестный автомат, $\lozenge$ – операция параллельной композиции, и $\cong$ – отношение эквивалентности. Как обычно, полностью определенный автомат B называется *решением* уравнения $C \lozenge X \cong S$, если $C \lozenge B \cong S$. Решение M уравнения называется *наибольшим*, если каждое решение уравнения есть редукция автомата M. Известно, что если уравнение $C \lozenge X \cong S$ разрешимо, то уравнение имеет наибольшее решение, которое в общем случае является недетерминированным частичным автоматом с языком $\overline{(L(C)\langle\rangle_{Ext}\overline{(L(S)\bigcap(IO)^*)})} \cap ((I_2 \cup U)(O_2 \cup V))^*$. В данной работе в качестве коэффициентов уравнения, автоматов C и S, и его решения мы рассматриваем только полностью определенные автоматы, поскольку любая физическая реализация является полностью определенной. Соответственно из наибольшего решения мы итеративно удаляем состояния, в которых не определен хотя бы один переход, и получаем наибольшее полностью определенное решение уравнения [1].

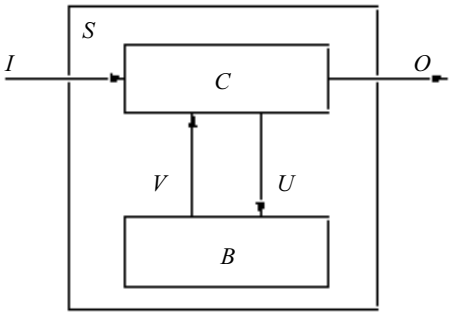


Рис. 1

В качестве примера рассмотрим композицию автоматов на рис. 1. Если поведение композиции описано автоматом $S = (\{ s\}, \{i\}, \{o\}, T_S, s)$, где $T_S = \{ (i, s, s, o)\}$, и фрагмент композиции, поведение которого описано автоматом $C = (\{ c\}, \{i, v\}, \{o, u\}, T_C, c)$, где $T_C = \{ (i, c, c, u), (v, c, c, o)\}$, уже синтезирован, то неизвестный фрагмент композиции можно синтезировать как автомат $B = (\{ b\}, \{ u\}, \{ v\}, T_B, b)$, где $T_B = \{ (u, s, s, v)\}$, который есть решение автоматного уравнения $C \lozenge X \cong S$.

[1] В выходной алфавит O композиции можно добавить символы из внутренних алфавитов U и V, но в данной работе без ограничения общности мы полагаем, что $O = O_1 \cup O_2$.

Решение Sol автоматного уравнения $C \Diamond X \cong S$ называется *прогрессивным*, если Sol полностью определенный автомат, и для любого слова α из языка $Init(L(C))_{\Uparrow(I_2 \cup O_2)} \cap Init(L(Sol))_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^{*\Uparrow(U \cup V)}$ существует продолжение в языке $(L(C)_{\Uparrow I2 \cup O2} \cap (L(Sol)_{\Uparrow I1 \cup O1} \cap (IO)^{*\Uparrow(U \cup V)}$. Таким образом, решение Sol называется прогрессивным, если из каждого состояния полуавтомата $C_{\Uparrow(I_2 \cup O_2)} \cap Sol_{\Uparrow(I_1 \cup O_1)} \cap (IO)^*$ достижимо финальное состояние, и в каждом финальном состоянии существует переход по каждому символу из алфавита I.

2. Алгоритм нахождения наибольшего прогрессивного решения

Постановка задачи. Даны полностью определенные автоматы $C = (C, I_1 \cup V, O_1 \cup U, T_C, c_0)$ и $S = (S, I_1 \cup I_2, O_1 \cup O_2, T_S, s_0)$ и разрешимое уравнение $C \Diamond X \cong S$. Необходимо найти наибольшую редукцию $Prog(Sol)$ наибольшего решения Sol уравнения, которая является прогрессивным решением. Если такой редукции не существует, то уравнение $C \Diamond X \cong S$ не имеет прогрессивных решений. Если такая редукция существует, то $Prog(Sol)$ есть наибольшее прогрессивное решение уравнения.

Алгоритм 1. Построение наибольшего прогрессивного решения (если прогрессивные решения существуют)

Вход: Полностью определенное наибольшее решение Sol уравнения $C \Diamond X \cong S$.

Выход: Наибольшая полностью определенная редукция $Prog(Sol)$ наибольшего решения, которая является прогрессивным решением (если прогрессивные решения существуют).

1. Положим j: = 1 и $P^j = Sol$.

2. Построим полуавтомат $R^j = C_{\Uparrow(I_2 \cup O_2)} \cap P^j{}_{\Uparrow(I_1 \cup O_1)} \cap (IO)^{*\Uparrow(U \cup V)}$ и положим R: = R^j.

3. Удалим в полуавтомате R все состояния (и переходы в эти состояния), в которых генерируется пустой язык. Если в полученном полуавтомате из каждого финального состояния есть переходы по всем внешним входным символам, то обозначаем полученный полуавтомат B^j и Шаг 5. Иначе Шаг 4.

4) Удаляем из B^j финальные состояния (и переходы в эти состояния), из которых нет перехода хотя бы для одного внешнего входного символа. Если удаляется начальное состояние, то прогрессивного решения не существует, КОНЕЦ. Иначе обозначаем полученный полуавтомат R и Шаг 3.

5) Находим проекцию полуавтомата $((Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^j)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^{*\Uparrow(U \cup V)}) \setminus Init(B^j))$ на алфавиты решения и вычитаем ее из полуавтомата P^j. Полученный полуавтомат представим автоматом P^{j+1}. Если автоматы P^{j+1} и P^j эквивалентны, то автомат P^j есть наибольшее прогрессивное решение, КОНЕЦ. Иначе в автомате P^{j+1} итеративно удалим состояния (и переходы в эти состояния), из которых нет перехода хотя бы для одного входного символа. Если удаляется начальное состояние, то прогрессивного решения не существует, КОНЕЦ. Иначе увеличиваем j на 1, и Шаг 2.

Теорема 1. Автомат, построенный по алгоритму 1, является наибольшим прогрессивным решением.

Доказательство. На Шаге 5 алгоритм заканчивает работу, если и только если из языка полуавтомата P^j не было удалено ни одной последовательности, т.е. P^j соответствует полностью определенному автомату, такому, что в полуавтомате $C_{\Uparrow(I_2 \cup O_2)} \cap P^j{}_{\Uparrow(I_1 \cup O_1)} \cap (IO)^{*\Uparrow(U \cup V)}$ отсутствуют состояния, из которых не достижимо финальное состояние, и из каждого финального состояния есть переходы по всем внешним входным символам, т.е. P^j есть прогрессивное решение.

Покажем, что для любого j, $j \geq 0$, язык L каждого прогрессивного решения уравнения содержится в языке P^{j+1} автомата P^{j+1}. По определению наибольшего решения, $L \subseteq P^1 = Sol$.

Предположим, что $L \subseteq P^j$ для некоторого $j \geq 1$. Покажем, что $L \subseteq P^{j+1}$. Так как $L \subseteq P^j$, то

$$R = Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(L)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^{*\Uparrow(U \cup V)} \subseteq Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^j)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^{*\Uparrow(U \cup V)}.$$

Пусть $Q^j = Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^j)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^{*\Uparrow(U \cup V)}$. Ниже представлены соотношения, которые показывают, что $L \cap (Q^j \setminus Init(B^j))_{\Downarrow(I_2 \cup O_2 \cup U \cup V)} = \varnothing$. Мы пользуемся равенством

$$(L_1 \cap L_2)_{\Downarrow(I_2 \cup U \cup V \cup O_2)} = L_{1\,\Downarrow(I_2 \cup U \cup V \cup O_2)} \cap L_{2\,\Downarrow(I_2 \cup U \cup V \cup O_2)},$$

так как язык $L_{\Uparrow(I_1 \cup O_1)}$ обладает свойством

$$((L_{\Uparrow(I_1 \cup O_1)})_{\Downarrow(I_2 \cup O_2 \cup U \cup V)})_{\Uparrow(I_1 \cup O_1)} = L_{\Uparrow(I_1 \cup O_1)}$$

и свойством

$$(L_{\Uparrow(I_1 \cup O_1)})_{\Downarrow(I_2 \cup O_2 \cup U \cup V)} = L \text{ [1].}$$

Таким образом,

$R \subseteq Init(B^j) \subseteq Q^j \Rightarrow$

$R \cap (Q^j \setminus Init(B^j)) = \varnothing \Rightarrow$

$(Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(L)_{\Uparrow(I_1 \cup O_1)}) \cap (Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^j)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^*_{\Uparrow(U \cup V)}) \cap \overline{Init(B^j)} = \varnothing \Rightarrow$

$Init(L)_{\Uparrow(I_1 \cup O_1)} \cap (Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^j)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^*_{\Uparrow(U \cup V)}) \cap \overline{Init(B^j)} = \varnothing \Rightarrow$

$L_{\Uparrow(I_1 \cup O_1)} \cap Q^j \cap \overline{Init(B^j)} = \varnothing \Rightarrow$

$L_{\Uparrow(I_1 \cup O_1)} \cap (Q^j \setminus Init(B^j)) = \varnothing.$

Так как $L \subseteq P^j$, $L \cap (Q^j \setminus Init(B^j))_{\Downarrow(I_2 \cup O_2 \cup U \cup V)} = \varnothing$ и $P^{j+1} = P^j \setminus (Q^j \setminus Init(B^j))_{\Downarrow(U \cup V \cup I_2 \cup O_2)}$, то $L \subseteq P^{j+1}$. ■

Теорема 2. Алгоритм 1 сходится.

Доказательство. Доказательство основывается на том факте, что язык каждого финального состояния полуавтомата R^j можно представить как конечное число объединений конечного числа пересечений языков (и их дополнений) состояний полуавтомата R^1, т.е. количество попарно не эквивалентных автоматов, т.е. автоматов с попарно различными языками, которое может быть сгенерировано на Шаге 5 алгоритма, является конечным. Поскольку на каждой итерации язык полуавтомата R^{j+1} строго содержится в языке полуавтомата R^j, то на некоторой итерации полуавтомат P^{j+1} будет эквивалентен полуавтомату P^j, т.е. алгоритм сходится.

Для простоты изложения мы только демонстрируем вышеизложенный факт на примере полуавтомата R^2, т.е. показываем, что язык каждого финального состояния R^2 может быть представлен как конечное число объединений конечного числа пересечений языков (и их дополнений) состояний полуавтомата R^1.

Рассмотрим языки $R^1 = C_{\Uparrow(I_2 \cup O_2)} \cap P^1_{\Uparrow(I_1 \cup O_1)}$ и $R^2 = C_{\Uparrow(I_2 \cup O_2)} \cap P^2_{\Uparrow(I_1 \cup O_1)}$, где C, P^1, P^2 есть соответственно языки полуавтоматов C, P^1, P^2. Пусть $Q^1 = Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^1)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^*_{\Uparrow(U \cup V)}$. Выразим R^2 через R^1:

$$
\begin{aligned}
R^2 &= C_{\Uparrow(I_2 \cup O_2)} \cap P^2_{\Uparrow(I_1 \cup O_1)} = \\
&= C_{\Uparrow(I_2 \cup O_2)} \cap [P^1 \setminus (Q^1 \setminus Init(B^1))_{\Downarrow(I_2 \cup U \cup V \cup O_2)}]_{\Uparrow(I_1 \cup O_1)} = \\
&= C_{\Uparrow(I_2 \cup O_2)} \cap [P^1 \cap \overline{(Q^1 \setminus Init(B^1))_{\Downarrow(I_2 \cup U \cup V \cup O_2)}}]_{\Uparrow(I_1 \cup O_1)} = \\
&= C_{\Uparrow(I_2 \cup O_2)} \cap P^1_{\Uparrow(I_1 \cup O_1)} \cap \overline{((Q^1 \setminus Init(B^1))_{\Downarrow(I_2 \cup U \cup V \cup O_2)})_{\Uparrow(I_1 \cup O_1)}} = \\
&= R^1 \cap \overline{((Q^1 \setminus Init(B^1))_{\Downarrow(I_2 \cup U \cup V \cup O_2)})_{\Uparrow(I_1 \cup O_1)}}.
\end{aligned}
$$

Заметим, что при построении R^2 мы вновь воспользовались равенством $(L_1 \cap L_2)_{\Uparrow(I_1 \cup O_1)} = L_{1\Uparrow(I_1 \cup O_1)} \cap L_{2\Uparrow(I_1 \cup O_1)}$.

Рассмотрим конструктивные шаги, ведущие от языка R^1 к языку R^2:

• Язык $(Q^1 \setminus Init(B^1))$ есть язык полуавтомата $M = (Init(C)_{\Uparrow(I_2 \cup O_2)} \cap Init(P^1)_{\Uparrow(I_1 \cup O_1)} \cap Init(IO)^*_{\Uparrow(U \cup V)}) \setminus Init(B^1)$, каждое финальное состояние которого представляет язык $L(r^1)$, который представлен состоянием r^1 полуавтомата R^1.

• Язык $(Q^1 \setminus Init(B^1))_{\Downarrow(I_2 \cup U \cup V \cup O_2)})_{\Uparrow(I_1 \cup O_1)}$ есть язык полуавтомата $M_{\Downarrow(U \cup V \cup I_2 \cup O_2)}$, каждое финальное состояние которого представляет язык $L(r^1)_{\Downarrow(I_2 \cup U \cup V \cup O_2)}$.

• Язык $R^1 \cap \overline{((Q^1 \setminus Init(B^1))_{\Downarrow(I_2 \cup U \cup V \cup O_2)})_{\Uparrow(I_1 \cup O_1)}}$ есть язык полуавтомата $R^1 \cap (\overline{M_{\Downarrow(U \cup V \cup I_2 \cup O_2)}})_{\Uparrow(I_1 \cup O_1)}$. Для того чтобы найти дополнение $\overline{M_{\Downarrow(U \cup V \cup I_2 \cup O_2)}}$, необходимо детерминизировать полуавтомат $M_{\Downarrow(U \cup V \cup I_2 \cup O_2)}$. Каждое состояние $\widehat{m}^1_j = \{m^1_h \in M \mid h \in H^1_j \subseteq J^1\}$ детерминированного полуавтомата будет представлять язык $L(\widehat{m}^1_j) = \bigcap_{h \in H^1_j \subseteq J^1} L(m^1_h)_{\Downarrow(U \cup V \cup I_2 \cup O_2)} = \bigcap_{h \in H^1_j \subseteq J^1} L(r^1_h)_{\Downarrow(U \cup V \cup I_2 \cup O_2)}$, где $J^1 = \{1,...,|M|\}$,

M – множество состояний полуавтомата $M_{\Downarrow(U\cup V\cup I_2\cup O_2)}$. Кроме того, к полуавтомату добавляется *безразличное* состояние $\{dc\}$, которое представляет язык $L(\{dc\})$, полученный дополнением объединения языков, представленных финальными состояниями детерминированного полуавтомата:

$$L(\{dc\}) = \overline{\bigcup_{j\in\hat{J}^1} L(\hat{m}_j^1)} = \bigcap_{j\in\hat{J}^1} \overline{L(\hat{m}_j^1)} = \bigcap_{j\in\hat{J}^1} \overline{\bigcap_{h\in H_j^1\subseteq J^1} L(r_h^1)_{\Downarrow(U\cup V\cup I_2\cup O_2)}} =$$

$$= \bigcap_{j\in\hat{J}^1} \bigcup_{h\in H_j^1\subseteq J^1} \overline{L(r_h^1)_{\Downarrow(U\cup V\cup I_2\cup O_2)}} = \bigcup_{K\in\mathcal{K}\subseteq 2^{J^1}} \bigcap_{k\in K} \overline{L(r_k^1)_{\Downarrow(U\cup V\cup I_2\cup O_2)}} \quad ,$$

где $\hat{J}^1 = \{1,...,\left|\widehat{M}\right|\}$ и $\widehat{M}$ есть множество состояний детерминированного полуавтомата.

Полуавтомат $\overline{M_{\Downarrow(U\cup V\cup I_2\cup O_2)}}$ получается из детерминированного полуавтомата переключением финальных и нефинальных состояний. Каждое финальное состояние $r_j^2 = (r_j^1, \hat{m}_p^1)$ полуавтомата $R^2 = R^1 \cap (\overline{M_{\Downarrow(U\cup V\cup I_2\cup O_2)}})_{\Uparrow(I_1\cup O_1)}$ представляет язык $L(r_j^2)$, который можно выразить в следующей форме: если $\hat{m}_p^1 \neq \{dc\}$, то

$$L(r_j^2) = L(r_j^1) \cap \bigcap_{h\in H_j^1\subseteq J^1} (L(r_h^1)_{\Downarrow(U\cup V\cup I_2\cup O_2)})_{\Uparrow(I_1\cup O_1)};$$

если $\hat{m}_p^1 = \{dc\}$, то

$$L(r_j^2) = L(r_j^1) \cap [\bigcup_{K\in\mathcal{K}\subseteq 2^{J^1}} \bigcap_{k\in K} \overline{L(r_k^1)_{\Downarrow(U\cup V\cup I_2\cup O_2)\Uparrow(I_1\cup O_1)}}] = \bigcup_{K\in\mathcal{K}\subseteq 2^{J^1}} [L(r_j^1) \cap \bigcap_{k\in K} \overline{L(r_k^1)_{\Downarrow(U\cup V\cup I_2\cup O_2)\Uparrow(I_1\cup O_1)}}]. \blacksquare$$

Таким образом, если наибольшее прогрессивное решение существует, то уравнение имеет прогрессивное решение, и более того, любое прогрессивное решение есть редукция наибольшего прогрессивного решения. Однако заметим, что, как показывает следующий пример, в общем случае не любая полностью определенная редукция наибольшего прогрессивного решения есть решение уравнения.

Рассмотрим уравнение $C \Diamond X \cong S$ для автоматов, представленных на рис. 2, *a* и *b*. Автомат, соответствующий наибольшему прогрессивному решению Sol, показан на рис. 2, *c*. Непосредственной проверкой нетрудно убедиться, что полностью определенная редукция наибольшего прогрессивного решения, показанная на рис. 2, *f*, не является прогрессивным решением уравнения. Полуавтомат, соответствующий композиции наибольшего прогрессивного решения и контекста, показан на рис. 2, *d*; полуавтомат, соответствующий композиции контекста и полностью определенной редукции Red наибольшего прогрессивного решения, показан на рис. 2, *e*.

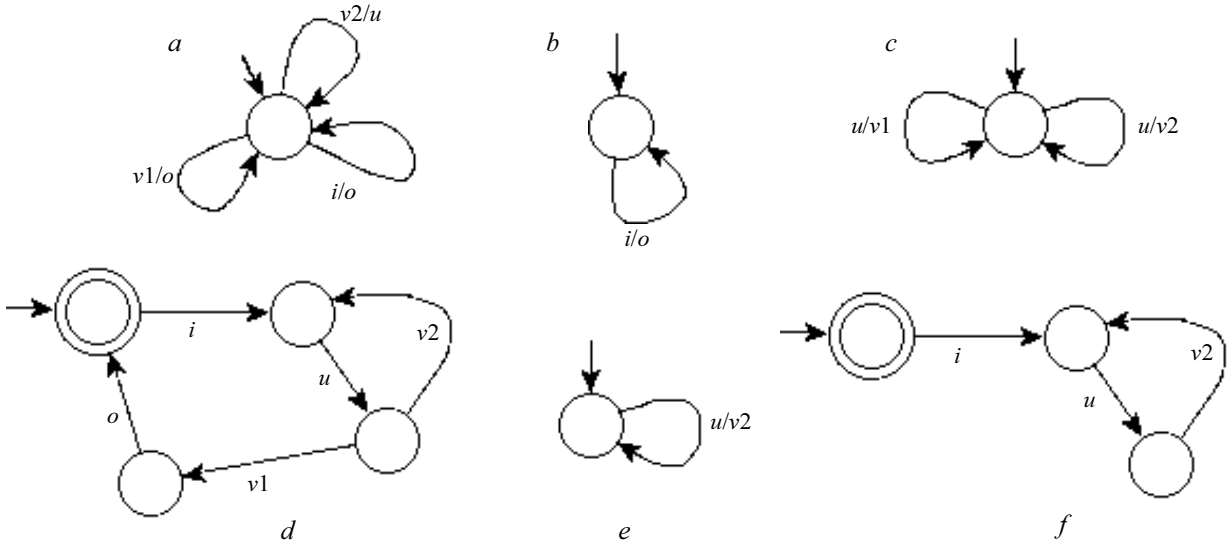


Рис. 2 Контекст C (*a*); спецификация S (*b*); наибольшее прогрессивное решение Sol (*c*); полуавтомат $C_{\Uparrow(I_2\cup O_2)} \cap Sol_{\Uparrow(I_1\cup O_1)}$ (*d*); полностью определенная редукция Red наибольшего прогрессивного решения, которая не является решением (*e*); полуавтомат $C_{\Uparrow(I_2\cup O_2)} \cap Red_{\Uparrow(I_1\cup O_1)}$ (*f*)

Заключение

В данной работе предложен алгоритм построения наибольшего прогрессивного решения параллельного автоматного уравнения, основанный на удалении из наибольшего решения последовательностей, которые ведут к возникновению заведомо тупиковых ситуаций в композиции с контекстом. Показана сходимость предложенного алгоритма. Показано также, что не каждая полностью определенная редукция наибольшего прогрессивного решения есть прогрессивное решение, и таким образом, проблема полной характеризации прогрессивных решений требует дополнительных исследований.

ЛИТЕРАТУРА

1. *Yevtushenko N.*, *Villa T.*, *Brayton R.K.*, *et al.* Sequential synthesis by language equation solving // International Workshop on Logic Synthesis. June, 2000.
2. *El-Fakih K.*, *Yevtushenko N.*, *Buffalov S.*, *Bochmann G.V.* Progressive solutions to a parallel automata equation // Theoretical Computer Science. October, 2006. P. 17 – 32.
3. *Yevtushenko N.*, *Villa T.*, *Brayton R.K.*, *et al.* Compositionally Progressive Solutions of Synchronous FSM Equation // Discrete Event Dynamic Systems. January, 2008. P. 51 – 89.

2008 Прикладная теория автоматов № 1(1)

DOI 10.17223/20710410/1/20

УДК 518.6+681.3

ХАРАКТЕРИСТИКА НЕПОДВИЖНЫХ ТОЧЕК ЛИНЕЙНЫХ АВТОМАТОВ НАД КОНЕЧНЫМ КОЛЬЦОМ

В.В. Скобелев

Институт прикладной математики и механики НАН Украины, г. Донецк

E-mail: vv_skobelev@iamm.ac.donetsk.ua

Исследуется структура множества неподвижных точек словарной функции, реализуемой инициальными линейными автоматами Мили и Мура над кольцом $\mathbf{Z}_{p^k}$. Охарактеризованы входные символы, являющиеся неподвижными точками для текущего состояния исследуемых автоматов.

Ключевые слова: *поточные шифры*, *линейные автоматы*, *конечные кольца*, *неподвижные точки.*

Нетривиальным обобщением линейных автоматов над конечным полем [1] являются линейные автоматы над кольцом $\mathbf{Z}_{p^k} = (\mathbf{Z}_{p^k}, \oplus, \circ)$ (p – простое число, $a \oplus b = a+b \pmod{p^k}$ и $a \circ b = a \cdot b \pmod{p^k}$). Сложность исследования таких автоматов обусловлена тем, что при переходе от поля к кольцу осуществляется переход от линейного пространства к модулю линейных форм [2]. В [3, 4] исследован ряд характеристик таких моделей с позиции теории автоматов и теории систем, а в [5] – применение этих моделей в качестве поточных шифров, при условии, что они являются БПИ-автоматами [6]. С этой позиции актуальным является исследование неподвижных точек словарных функций [7, 8], реализуемых инициальными линейными автоматами над кольцом $\mathbf{Z}_{p^k}$. Действительно, неподвижной точкой словарной функции $f: X^+ \to X^+$ называется любая такая последовательность $u \in X^+$, что истинно равенство $f(u) = u$. Поэтому именно неподвижная точка представляет собой «открытый текст», который не изменяется в процессе шифрования. Исследование структуры множества неподвижных точек линейных автоматов над кольцом $\mathbf{Z}_{p^k}$ и является основной целью настоящей работы.

Структура работы следующая: в п. 1 введены основные понятия; в п. 2 охарактеризовано множество неподвижных точек словарной функции, реализуемой инициальными линейными автоматами Мили и Мура над кольцом $\mathbf{Z}_{p^k}$; в п. 3 охарактеризованы входные символы, являющиеся неподвижными точками для текущего состояния исследуемых автоматов. Заключение содержит ряд выводов.

1. Основные понятия

Объектом исследования являются линейные автоматы Мили и Мура над кольцом $\mathbf{Z}_{p^k}$ соответственно

$$M_1: \begin{cases} \boldsymbol{q}_{t+1} = A \circ \boldsymbol{q}_t \oplus B \circ \boldsymbol{x}_{t+1}, \\ \boldsymbol{y}_{t+1} = C \circ \boldsymbol{q}_t \oplus D \circ \boldsymbol{x}_{t+1}, \end{cases} \quad (t \in \mathbf{Z}_+) \tag{1}$$

и

$$M_2: \begin{cases} \boldsymbol{q}_{t+1} = A \circ \boldsymbol{q}_t \oplus B \circ \boldsymbol{x}_{t+1}, \\ \boldsymbol{y}_{t+1} = C \circ \boldsymbol{q}_{t+1}, \end{cases} \quad (t \in \mathbf{Z}_+), \tag{2}$$

где n – фиксированное число; A, B, C, D – $(n \times n)$-матрицы над кольцом $\mathbf{Z}_{p^k}$, а $\boldsymbol{q}_t$, $\boldsymbol{x}_t$, $\boldsymbol{y}_t \in \mathbf{Z}_{p^k}^n$ – вектор-столбцы, соответствующие соответственно состоянию автомата, входному символу и выходному символу в момент t. Обозначим через A_1 и A_2 множества всех автоматов соответственно M_1 и M_2. При фиксации начального состояния $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$ автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ получим инициальный автомат $(M, \boldsymbol{q}_0)$.

Обозначим через $S_{fxd}(M, \boldsymbol{q}_0)$ $(M \in \mathsf{A}_1 \cup \mathsf{A}_2, \boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n)$ множество всех неподвижных точек словарной функции $f_{(M,\boldsymbol{q}_0)} : (\mathbf{Z}_{p^k}^n)^+ \to (\mathbf{Z}_{p^k}^n)^+$, реализуемой инициальным автоматом $(M, \boldsymbol{q}_0)$. Положим

$$S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0) = S_{fxd}(M, \boldsymbol{q}_0) \cap (\mathbf{Z}_{p^k}^n)^{t+1} \quad (t \in \mathbf{Z}_+).$$

Для любого инициального автомата $(M, \boldsymbol{q}_0)$ истинно равенство

$$S_{fxd}(M, \boldsymbol{q}_0) = \bigcup_{t=0}^{\infty} S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0). \tag{3}$$

При этом если $t_1 \neq t_2$, то

$$S_{fxd}^{(t_1+1)}(M, \boldsymbol{q}_0) \cap S_{fxd}^{(t_2+1)}(M, \boldsymbol{q}_0) = \varnothing.$$

Поэтому для исследования структуры множества $S_{fxd}(M, \boldsymbol{q}_0)$ ($M \in \mathsf{A}_1 \cup \mathsf{A}_2$, $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$) достаточно охарактеризовать общий элемент последовательности $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0)$ ($t \in \mathbf{Z}_+$).

Отметим, что из включения $S_{fxd}^{(t+2)}(M, \boldsymbol{q}_0) \subseteq S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0) \cdot \mathbf{Z}_{p^k}^n$ ($t \in \mathbf{Z}_+$) вытекает

Утверждение 1. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ при любом начальном состоянии $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$, если существует такое значение $t_0 \in \mathbf{Z}_+$, что $S_{fxd}^{(t_0+1)}(M, \boldsymbol{q}_0) = \varnothing$, то $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0) = \varnothing$ для всех $t > t_0$.

Из утверждения 1 и равенства (3), в свою очередь, вытекает

Утверждение 2. Множество $S_{fxd}(M, \boldsymbol{q}_0)$ ($M \in \mathsf{A}_1 \cup \mathsf{A}_2$, $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$) – конечное тогда и только тогда, когда существует такое значение $t_0 \in \mathbf{Z}_+$, что $S_{fxd}^{(t_0+1)}(M, \boldsymbol{q}_0) = \varnothing$.

2. Характеристика множества $S_{fxd}(M, \boldsymbol{q}_0)$ ($M \in \mathsf{A}_1 \cup \mathsf{A}_2$, $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$)

Выразим $\boldsymbol{y}_{t+1}$ ($t \in \mathbf{N}$) из систем (1) и (2) через начальное состояние $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$ автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ и элементы входной последовательности $\boldsymbol{x}_1 \ldots \boldsymbol{x}_{t+1} \in (\mathbf{Z}_{p^k}^n)^{t+1}$. Получим

$$\boldsymbol{y}_{t+1} = C \circ (A^t \circ \boldsymbol{q}_0 \oplus (\bigoplus_{j=1}^{t-1} A^{t-j} \circ B \circ \boldsymbol{x}_j) \oplus B \circ \boldsymbol{x}_t) \oplus D \circ \boldsymbol{x}_{t+1} \ (t \in \mathbf{N}), \tag{4}$$

если $M \in \mathsf{A}_1$ и

$$\boldsymbol{y}_{t+1} = C \circ (A^{t+1} \circ \boldsymbol{q}_0 \oplus (\bigoplus_{j=0}^{t-1} A^{t-j} \circ B \circ \boldsymbol{x}_{j+1}) \oplus B \circ \boldsymbol{x}_{t+1}) \ (t \in \mathbf{N}), \tag{5}$$

если $M \in \mathsf{A}_2$.

Из (1), (2), (4) и (5) вытекает

Теорема. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ при любом начальном состоянии $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$ для всех $t \in \mathbf{Z}_+$ множество $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0)$ состоит из всех таких слов $\boldsymbol{x}_1 \ldots \boldsymbol{x}_{t+1} \in (\mathbf{Z}_{p^k}^n)^{t+1}$, что:

а) если $M \in \mathsf{A}_1$, то $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t+1})$ – множество решений системы уравнений

$$\begin{cases} (I \ominus D) \circ \boldsymbol{x}_1 = C \circ \boldsymbol{q}_0, \\ (I \ominus D) \circ \boldsymbol{x}_{i+1} = C \circ (A^i \circ \boldsymbol{q}_0 \oplus (\bigoplus_{j=1}^{i-1} A^{i-j} \circ B \circ \boldsymbol{x}_j) \oplus B \circ \boldsymbol{x}_i) \quad (i = 1, \ldots, t), \end{cases} \tag{6}$$

б) если $M \in \mathsf{A}_2$, то $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t+1})$ – множество решений системы уравнений

$$\begin{cases} (I \ominus C \circ B) \circ \boldsymbol{x}_1 = C \circ A \circ \boldsymbol{q}_0, \\ (I \ominus C \circ B) \circ \boldsymbol{x}_{i+1} = C \circ A \circ (A^i \circ \boldsymbol{q}_0 \oplus \bigoplus_{j=1}^{i-1} A^{i-j} \circ B \circ \boldsymbol{x}_j \oplus B \circ \boldsymbol{x}_i) \quad (i = 1, \ldots, t). \end{cases} \tag{7}$$

Отметим, что каждое уравнение систем (6) и (7) имеет вид

$$A \circ \boldsymbol{x} = \boldsymbol{b}. \tag{8}$$

Известно (см., напр., [9]), что для уравнения (8) возможна одна из следующих трех ситуаций:

1) уравнение (8) не имеет решений;

2) уравнение (8) имеет единственное решение;

3) число решений уравнения (8) равно p^r, где $r \in \{1, \ldots, kn\}$.

Именно эти утверждения и характеризуют число решений систем уравнений (6) и (7).

Рассмотрим ряд следствий из теоремы.

Следствие 1. Для любого автомата $M \in \mathsf{A}_1$, если матрица $I \ominus D$ – обратимая, то при любом начальном состоянии $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$ множество $S_{fxd}(M, \boldsymbol{q}_0)$ – бесконечное, причем множество $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0)$ – одноэлементное для каждого $t \in \mathbf{Z}_+$ и имеет вид $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0) = \{\boldsymbol{x}_1 \ldots \boldsymbol{x}_{t+1}\}$, где

$$\begin{cases} \boldsymbol{x}_1 = (I \ominus D)^{-1} \circ C \circ \boldsymbol{q}_0, \\ \boldsymbol{x}_{i+1} = (I \ominus D)^{-1} \circ C \circ (A^i \circ \boldsymbol{q}_0 \oplus (\overset{i-1}{\underset{j=1}{\oplus}} A^{i-j} \circ B \circ \boldsymbol{x}_j) \oplus B \circ \boldsymbol{x}_i) \quad (i = 1, ..., t). \end{cases}$$

Следствие 2. Для любого автомата $M \in \mathsf{A}_2$, если матрица $I \ominus C \circ B$ – обратимая, то при любом начальном состоянии $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$ множество $S_{fxd}(M, \boldsymbol{q}_0)$ – бесконечное, причем множество $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0)$ – одноэлементное для каждого $t \in \mathbf{Z}_+$ и имеет вид $S_{fxd}^{(t+1)}(M, \boldsymbol{q}_0) = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t+1}\}$, где

$$\begin{cases} \boldsymbol{x}_1 = (I \ominus C \circ B)^{-1} \circ C \circ A \circ \boldsymbol{q}_0, \\ \boldsymbol{x}_{i+1} = (I \ominus C \circ B)^{-1} \circ C \circ A \circ (A^i \circ \boldsymbol{q}_0 \oplus \overset{i-1}{\underset{j=1}{\oplus}} A^{i-j} \circ B \circ \boldsymbol{x}_j \oplus B \circ \boldsymbol{x}_i) \quad (i = 1, \ldots, t). \end{cases}$$

Из 1-го уравнения систем (6) и (7) вытекает, что для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ при любом начальном состоянии $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$ существует следующий локальный критерий проверки пустоты множества $S_{fxd}(M, \boldsymbol{q}_0)$.

Следствие 3. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ множество $S_{fxd}(M, \boldsymbol{q}_0)$ – непустое для таких и только таких состояний $\boldsymbol{q}_0 \in \mathbf{Z}_{p^k}^n$, для которых имеет решения уравнение

$$(I \ominus D) \circ \boldsymbol{x} = C \circ \boldsymbol{q}_0 \ (\boldsymbol{x} \in \mathbf{Z}_{p^k}^n), \tag{9}$$

если $M \in \mathsf{A}_1$, и уравнение

$$(I \ominus C \circ B) \circ \boldsymbol{x} = C \circ A \circ \boldsymbol{q}_0 \ (\boldsymbol{x} \in \mathbf{Z}_{p^k}^n), \tag{10}$$

если $M \in \mathsf{A}_2$.

Если $\boldsymbol{q}_0 = \mathbf{0}$, то и (9) и (10) имеют решение $\boldsymbol{x}_1 = \mathbf{0}$. Отсюда вытекает

Следствие 4. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$

$$S_{fxd}(M, \mathbf{0}) \neq \varnothing.$$

3. Характеристика множества $S_{fxd}^{(1)}(M, q)$ ($M \in \mathsf{A}_1 \cup \mathsf{A}_2$, $q \in \mathbf{Z}_{p^k}^n$)

Особенность следствия 3 состоит в том, что оно представляет собой «локальный критерий», т.е. характеристику структуры всего множества $S_{fxd}(M, \boldsymbol{q}_0)$, являющегося подмножеством свободной полугруппы $(\mathbf{Z}_{p^k}^n)^+$, в терминах образующего элемента $\boldsymbol{x} \in \mathbf{Z}_{p^k}^n$ полугруппы, выраженных через структуру множества $S_{fxd}^{(1)}(M, \boldsymbol{q}_0)$. Отсюда вытекает целесообразность исследования структуры множества $S_{fxd}(M, \boldsymbol{q})$ для $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ при любом текущем состоянии $\boldsymbol{q} \in \mathbf{Z}_{p^k}^n$. Рассмотрим некоторые такие характеристики.

Из следствия 3 вытекает

Следствие 5. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ проверка пустоты множества $S_{fxd}(M, \boldsymbol{q})$ при любом текущем состоянии $\boldsymbol{q} \in \mathbf{Z}_{p^k}^n$ сводится к проверке пустоты множества решений уравнения

$$(I \ominus D) \circ \boldsymbol{x} = C \circ \boldsymbol{q} \ (\boldsymbol{x} \in \mathbf{Z}_{p^k}^n), \tag{11}$$

если $M \in \mathsf{A}_1$, и уравнения

$$(I \ominus C \circ B) \circ \boldsymbol{x} = C \circ A \circ \boldsymbol{q} \ (\boldsymbol{x} \in \mathbf{Z}_{p^k}^n), \tag{12}$$

если $M \in \mathsf{A}_2$.

Из равенств (11) и (12) вытекает

Следствие 6. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ и для любых состояний $\boldsymbol{q}', \boldsymbol{q}'' \in \mathbf{Z}_{p^k}^n$, если $\boldsymbol{x}' \in S_{fxd}^{(1)}(M, \boldsymbol{q}')$ и $\boldsymbol{x}''' \in S_{fxd}^{(1)}(M, \boldsymbol{q}'')$, то $\boldsymbol{x}' \ominus \boldsymbol{x}''' \in S_{fxd}^{(1)}(M, \boldsymbol{q}' \ominus \boldsymbol{q}'')$.

Множество $S_{fxd}(M, \mathbf{0})$ ($M \in \mathsf{A}_1 \cup \mathsf{A}_2$) следующим образом характеризует множество $S_{fxd}(M, \boldsymbol{q})$ для любого текущего состояния $\boldsymbol{q} \in \mathbf{Z}_{p^k}^n$.

Следствие 7. Для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ при каждом текущем состоянии $\boldsymbol{q} \in \mathbf{Z}_{p^k}^n$ для любого входного символа $\boldsymbol{x}' \in S_{fxd}^{(1)}(M, \boldsymbol{q})$ истинно равенство

$$S^{(1)}_{fxd}\ (M, \boldsymbol{q}) = \{\, \boldsymbol{x}' \oplus\ \boldsymbol{x}'' \mid \boldsymbol{x}'' \in\ S^{(1)}_{fxd}\ (M, \mathbf{0})\}. \tag{13}$$

Из следствия 7 вытекает, что для любого автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ в явном виде достаточно построить только множество $S^{(1)}_{fxd}\ (M, \mathbf{0})$. Для любого текущего состояния $\boldsymbol{q} \in \mathbf{Z}^n_{p^k}$ множество $S^{(1)}_{fxd}\ (M, \boldsymbol{q})$ всегда может быть вычислено в соответствии с равенством (13).

Рассмотрим теперь автомат $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ специального вида.

Из (11) и (12) вытекает

Следствие 8. Пусть матрицы, определяющие автомат $M \in \mathsf{A}_1 \cup \mathsf{A}_2$, выбраны так, что:

а) $D = I$, если $M \in \mathsf{A}_1$;

б) $C \circ B = I$, если $M \in \mathsf{A}_2$.

Тогда

$$S^{(1)}_{fxd}\ (M, \boldsymbol{q}) = \mathbf{Z}^n_{p^k}$$

для любого такого текущего состояния $\boldsymbol{q} \in \mathbf{Z}^n_{p^k}$, что имеет решение уравнение

$$C \circ \boldsymbol{q} = \mathbf{0},$$

если $M \in \mathsf{A}_1$, и уравнение

$$C \circ A \circ \boldsymbol{q} = \mathbf{0},$$

если $M \in \mathsf{A}_2$.

Пусть матрицы, определяющие автомат $M \in \mathsf{A}_1 \cup \mathsf{A}_2$, выбраны так, что:

а) $I \ominus D = \alpha \circ C$, если $M \in \mathsf{A}_1$;

б) $I \ominus C \circ B = \alpha \circ C \circ A$, если $M \in \mathsf{A}_2$,

α – фиксированный элемент кольца $\mathbf{Z}_{p^k}$. Тогда уравнения (11) и (12) принимают соответственно вид

$$C \circ (\alpha \circ \boldsymbol{x}) = C \circ \boldsymbol{q}\ (\boldsymbol{x} \in \mathbf{Z}^n_{p^k}) \tag{14}$$

и

$$C \circ A\ (\alpha \circ \boldsymbol{x}) = C \circ A \circ \boldsymbol{q}\ (\boldsymbol{x} \in \mathbf{Z}^n_{p^k}). \tag{15}$$

Из (14) (15) вытекает

Следствие 9. Пусть существует такой элемент α кольца $\mathbf{Z}_{p^k}$, что для автомата $M \in \mathsf{A}_1 \cup \mathsf{A}_2$ имеет место равенство $I \ominus D = \alpha \circ C$, если $M \in \mathsf{A}_1$, и равенство $I\ \ominus C \circ B = \alpha \circ C \circ A$, если $M \in \mathsf{A}_2$. Тогда

$$S^{(1)}_{fxd}\ (M, \boldsymbol{q}) \neq \varnothing$$

для любого такого текущего состояния $\boldsymbol{q} \in \mathbf{Z}^n_{p^k}$, для которого имеет решение уравнение

$$\alpha \circ \boldsymbol{x} = \boldsymbol{q}\ (\boldsymbol{x} \in\ \mathbf{Z}^n_{p^k}). \tag{16}$$

Заключение

В работе исследована структура множества $S_{fxd}\ (M, \boldsymbol{q}_0)$ неподвижных точек словарной функции, реализуемой инициальными линейными автоматами $(M, \boldsymbol{q}_0)$ Мили и Мура над кольцом $\mathbf{Z}_{p^k}$. Установлен критерий, когда множество $S_{fxd}\ (M, \boldsymbol{q}_0)$ – непустое, а также найдены условия, при которых $S_{fxd}\ (M, \boldsymbol{q}_0)$ – бесконечное множество. Охарактеризованы входные символы, являющиеся неподвижными точками для текущего состояния исследуемых автоматов. Показано, что в явном виде достаточно хранить только множество входных символов, являющихся неподвижными точками для состояния $\boldsymbol{q} = \mathbf{0}$ исследуемого автомата, а множество входных символов, являющихся неподвижными точками для любого текущего состояния $\boldsymbol{q}$, достаточно легко может быть вычислено по этому множеству. Таким образом, в процессе подачи на исследуемый автомат входного слова $\boldsymbol{x}_1 \ldots \boldsymbol{x}_{t+1} \in (\mathbf{Z}^n_{p^k})^{t+1}$ $(t \in \mathbf{Z}_+)$ определение тех элементов выходного слова $\boldsymbol{y}_1 \ldots \boldsymbol{y}_{t+1} \in (\mathbf{Z}^n_{p^k})^{t+1}$, которые совпадают с соответствующими входными символами, может быть сведено к локальным действиям. Действительно, если $\boldsymbol{q} \in \mathbf{Z}^n_{p^k}$ – текущее состояние $\boldsymbol{q} \in \mathbf{Z}^n_{p^k}$ автомата M, $\boldsymbol{x} \in \mathbf{Z}^n_{p^k}$ – текущий входной символ, то проверка равенства входного символа $\boldsymbol{x}$ соответствующему выходному символу сводится к проверке равенства (11) для автомата Мили и равенства (12) для автомата Мура.

Полученные в работе результаты имеют следующее естественное обобщение. Зафиксируем обратимый элемент $\gamma \in \mathbf{Z}_{p^k}^n$ кольца $\mathbf{Z}_{p^k}$. Назовем γ-неподвижной точкой словарной функции $f_{(M,\boldsymbol{q}_0)}: (\mathbf{Z}_{p^k}^n)^+ \to (\mathbf{Z}_{p^k}^n)^+$ любое такое входное слово $\boldsymbol{x}_1 \ldots \boldsymbol{x}_{t+1} \in (\mathbf{Z}_{p^k}^n)^{t+1}$ $(t \in \mathbf{Z}_+)$, что

$$f_{(M,\boldsymbol{q}_0)}(\boldsymbol{x}_1 \ldots \boldsymbol{x}_{t+1}) = (\gamma \circ \boldsymbol{x}_1) \ldots (\gamma \circ \boldsymbol{x}_{t+1}).$$

Если под множеством $S_{fxd}(M, \boldsymbol{q})$ понимать множество всех γ-неподвижных точек словарной функции $f_{(M,\boldsymbol{q}_0)}$, то все полученные в работе результаты остаются истинными, если матрицу I заменить матрицей $\gamma \circ I$.

ЛИТЕРАТУРА

1. *Гилл А*. Линейные последовательностные машины. М.: Наука, 1974. 288 с.
2. *Ван дер Варден Б.Л.* Алгебра. М.: Наука, 1979. 624 с.
3. *Скобелев В.В.* Анализ линейных автоматов над кольцом $\mathbf{Z}_{p^k}$ // Труды института прикладной математики и механики НАН Украины. 2007. Т. 14. С. 162 – 173.
4. *Скобелев В.В.* Задача идентификации линейных автоматов над кольцом $\mathbf{Z}_{p^k}$ // Труды VII Междунар. конф. «Идентификация систем и задачи управления (SICPRO'08)» (Москва, 28 – 31 января 2008 г.). М.: ИПУ РАН, 2008. С. 1154 – 1185.
5. *Скобелев В.В.* Шифры на основе линейных БПИ-автоматов над кольцом $\mathbf{Z}_{p^k}$ // Вестник ТГУ. Приложение. 2007. № 23. С. 118 – 122.
6. *Курмит А.А.* Автоматы без потери информации конечного порядка. Рига: Зинатне, 1972. 266 с.
7. *Кудрявцев В.Б. и др*. Введение в теорию конечных автоматов. М.: Наука, 1985. 320 с.
8. *Трахтенброт Б.А.*, *Барздинь Я.М.* Конечные автоматы (поведение и синтез). М.: Наука, 1970. 400 с.
9. *Скобелев В.В.* Об обратимых матрицах над кольцом $\mathbf{Z}_{p^k}$ // Труды института прикладной математики и механики НАН Украины. 2006. Вып. 13. С. 185 – 192.

2008 Прикладная теория автоматов № 1(1)
DOI 10.17223/20710410/1/21
УДК 681.3

ИДЕНТИФИКАЦИЯ АВТОМАТА В КЛАССЕ АВТОМАТОВ СПРОТТА

В.А. Сухинин, В.Г. Скобелев

Донецкий национальный университет
Институт прикладной математики и механики НАН Украины, г. Донецк

E-mail: vladimir.suhinin@gmail.com, skbv@iamm.ac.donetsk.ua

Решается задача идентификации автомата в классе автоматов Спротта над конечным кольцом $\mathbf{Z}_p{}^k = (\mathbf{Z}_p{}^k, \oplus, \circ)$. Оценка сложности решения задачи необходима для характеристики стойкости поточного шифра, определяемого автоматом Спротта при использовании информационного потока в качестве управления. Показано, что в подклассе автоматов Спротта задача решается проведением кратного эксперимента с автоматом.

Ключевые слова: *автоматы Спротта*, *идентификация*, *кратные эксперименты*, *поточные шифры*, *криптоанализ.*

На современном этапе развития информационных технологий актуальной проблемой является построение высокоскоростных поточных шифров. Значительные усилия исследователей направлены на разработку математического аппарата, предназначенного для решения задач преобразования информации, основанного на моделях и методах современной алгебры. Одним из таких направлений (см., напр., [1, 2]) является разработка средств и методов защиты информации на основе хаотических динамических систем [3]. При использовании информационного потока в качестве управления любая такая система определяет симметричный поточный шифр, причем расшифрование осуществляет обратная система. Ошибки округления, возникающие при использовании таких шифров над полем действительных чисел, устраняются за счет перехода к конечному кольцу $\mathbf{Z}_p{}^k = (\mathbf{Z}_p{}^k, \oplus, \circ)$. Такой переход приводит к конечным автоматам над конечным кольцом, что дает возможность непосредственно применять при анализе шифров модели и методы теории автоматов и теории колец. При этом параметры автомата и его начальное состояние являются составляющими секретного ключа шифра.

Структура работы следующая: в п. 1 представлена основная модель – аналоги над конечным кольцом систем Спротта с управлением; в п. 2 решена задача идентификации автомата в подклассе автоматов Спротта. Заключение содержит ряд выводов.

1. Основная модель

В дальнейшем все величины – элементы кольца $\mathbf{Z}_p{}^k = (\mathbf{Z}_p{}^k, \oplus, \circ)$, где p – простое число, $k \in \mathbf{N}$, а операции $\oplus$ и $\circ$ определяются равенствами

$$a \oplus b = a + b \pmod{p^k},$$
$$a \circ b = a \cdot b \pmod{p^k},$$

где $a, b \in \mathbf{Z}_p{}^k$, $\ominus$ – операция, обратная к операции $\oplus$, причем будем считать, что $p^k > 2$.

В работе исследуется класс автоматов Спротта – класс аналогов модельных динамических систем Спротта с нетривиальной структурой множества аттракторов [3]. Каждая из классических систем Спротта представляет собой систему трех дифференциальных уравнений. Поэтому информационная переменная может быть добавлена как во все уравнения системы (тогда выходом автомата является состояние автомата), так и в некоторые уравнения системы (соответственно выходом автомата является частичная информация о его состоянии). Положим $\mathrm{Y} = \{A, B, \ldots, S\}$. Автомат, соответствующий системе Спротта $x \in \mathrm{Y}$ в случае, когда информационная переменная добавляется во все уравнения системы, обозначим через M_x. В табл. 1 представлены автоматы M_x ($x \in \mathrm{Y}$) в скалярном виде, т.е. в виде систем уравнений над кольцом $\mathbf{Z}_p{}^k$, где $m_i = h \circ \alpha_i \circ x_{t+1}$ ($i = 1, 2, 3$), $t \in \mathbf{Z}_0$, а h и α_i ($i = 1, 2, 3$) – обратимые элементы кольца $\mathbf{Z}_p{}^k$. При переходе к кольцу $\mathbf{Z}_p{}^k$ ряд систем Спротта обобщен. Это обобщение состоит в том, что конкретные числовые коэффициенты классических систем Спротта заменены на произвольный фиксированный элемент β (возможно, с ин-

дексами) кольца $\mathbf{Z}_p^k$. Кроме того, в табл. 1 не указаны выходные переменные автоматов, так как для автоматов M_x ($x \in \mathrm{Y}$) выход автомата совпадает с его состоянием.

Таблица 1

$(M_A, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \ominus h \circ q_t^{(1)} \oplus h \circ q_t^{(2)} \circ q_t^{(3)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \ominus h \circ (q_t^{(2)})^2 \oplus m_3 \end{cases}$	$(M_B, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \ominus h \circ q_t^{(1)} \circ q_t^{(2)} \oplus m_3 \end{cases}$
$(M_C, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \ominus h \circ (q_t^{(1)})^2 \oplus m_3 \end{cases}$	$(M_D, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \oplus h \circ q_t^{(3)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \circ q_t^{(3)} \oplus \\ \quad \oplus \beta \circ h \circ (q_t^{(2)})^2 \oplus m_3 \end{cases}$
$(M_E, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ (q_t^{(1)})^2 \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \ominus h \circ \beta \circ q_t^{(1)} \oplus m_3 \end{cases}$	$(M_F, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus h \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \ominus h \circ q_t^{(1)} \oplus h \circ \beta \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ (q_t^{(1)})^2 \ominus h \circ q_t^{(3)} \oplus m_3 \end{cases}$
$(M_G, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ \beta \circ q_t^{(1)} \oplus h \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \circ q_t^{(3)} \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \ominus h \circ q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus m_3 \end{cases}$	$(M_H, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ q_t^{(2)} \oplus h \circ (q_t^{(3)})^2 \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \oplus h \circ \beta \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \ominus h \circ q_t^{(3)} \oplus m_3 \end{cases}$
$(M_I, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ \beta \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \oplus h \circ q_t^{(3)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \oplus h \circ (q_t^{(2)})^2 \ominus \\ \quad \ominus h \circ q_t^{(3)} \oplus m_3 \end{cases}$	$(M_J, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ \beta \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \ominus h \circ \beta \circ q_t^{(2)} \oplus h \circ q_t^{(3)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \ominus h \circ q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus \\ \quad \oplus h \circ (q_t^{(2)})^2 \oplus m_3 \end{cases}$
$(M_K, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(1)} \circ q_t^{(2)} \ominus h \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \oplus h \circ \beta \circ q_t^{(3)} \oplus m_3 \end{cases}$	$(M_L, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus h \circ \beta_1 \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ \beta_2 \circ (q_t^{(1)})^2 \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \ominus h \circ q_t^{(1)} \oplus m_3 \end{cases}$
$(M_M, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \ominus h \circ (q_t^{(1)})^2 \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ \beta \oplus h \circ \beta \circ q_t^{(1)} \oplus \\ \quad \oplus h \circ (q_t^{(2)})^2 \oplus m_3 \end{cases}$	$(M_N, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ \beta \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \oplus h \circ (q_t^{(3)})^2 \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \oplus h \circ q_t^{(2)} \ominus h \circ \beta \circ q_t^{(1)} \oplus m_3 \end{cases}$
$(M_O, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \ominus h \circ q_t^{(3)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \oplus h \circ q_t^{(1)} \circ q_t^{(3)} \oplus \\ \quad \oplus h \circ \beta \circ q_t^{(2)} \oplus m_3 \end{cases}$	$(M_P, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ \beta \circ q_t^{(2)} \oplus h \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \ominus h \circ q_t^{(1)} \oplus h \circ (q_t^{(2)})^2 \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \oplus h \circ q_t^{(2)} \oplus m_3 \end{cases}$
$(M_Q, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ q_t^{(3)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \ominus h \circ q_t^{(2)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ \beta_1 \oplus h \circ (q_t^{(2)})^2 \oplus \\ \quad \oplus h \circ \beta_2 \circ q_t^{(3)} \oplus m_3 \end{cases}$	$(M_R, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \oplus h \circ \beta_1 \ominus h \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ \beta_2 \oplus h \circ q_t^{(3)} \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \circ q_t^{(1)} \circ q_t^{(2)} \ominus h \circ q_t^{(3)} \oplus m_3 \end{cases}$
$(M_S, \mathrm{q}_0): \begin{cases} q_{t+1}^{(1)} = q_t^{(1)} \ominus h \circ q_t^{(1)} \ominus h \circ \beta \circ q_t^{(2)} \oplus m_1 \\ q_{t+1}^{(2)} = q_t^{(2)} \oplus h \circ q_t^{(1)} \oplus h \circ (q_t^{(3)})^2 \oplus m_2 \\ q_{t+1}^{(3)} = q_t^{(3)} \oplus h \oplus h \circ q_t^{(1)} \oplus m_3 \end{cases}$	

2. Идентификация автомата в классе автоматов M_x ($x \in Y$)

При атаках на шифрсистему, основанную на использовании автоматов M_x ($x \in Y$), важной информацией для криптоаналитика является знание x, то есть знание автомата, с которым работает шифрсистема. Предположим, что криптоаналитик может осуществлять требуемую инициализацию $(q_0^{(1)}, q_0^{(2)}, q_0^{(3)}) \in (\mathbf{Z}_{p^k})^3$ автомата M_x ($x \in Y$), а также имеет возможность управлять входом автомата M_x ($x \in Y$) и наблюдать его реакцию на поданную входную последовательность. Истинна следующая теорема.

Теорема 1. Автомат M_x ($x \in Y$) можно идентифицировать 4-кратным экспериментом высоты 1.

Доказательство. Каждый автомат M_x ($x \in Y$) установим в одно из состояний $q_0 = (1, 0, 0)$, $q_0 = (0, 1, 0)$, $q_0 = (0, 0, 1)$ либо $q_0 = (0, 0, 0)$. В каждом из этих состояний подадим на автомат входной символ 0. Полученные выходы автоматов занесем в табл. 2.

Таблица 2

Автомат	$q_0 = (1,0,0)$	$q_0 = (0,1,0)$	$q_0 = (0,0,1)$	$q_0 = (0,0,0)$
(M_A,q_0)	$(1,\ominus h,h)$	$(h,1,0)$	$(0,0,1\oplus h)$	$(0,0,h)$
(M_B,q_0)	$(1,h,h)$	$(0,1\ominus h,h)$	$(0,0,1\oplus h)$	$(0,0,h)$
(M_C,q_0)	$(1,h,0)$	$(0,1\ominus h,h)$	$(0,0,1\oplus h)$	$(0,0,h)$
(M_D,q_0)	$(1,h,0)$	$(\ominus h,1,h\circ\beta)$	$(0,h,1)$	$(0,0,0)$
(M_E,q_0)	$(1,h,h\ominus h\circ\beta)$	$(0,1\ominus h,h)$	$(0,0,1\oplus h)$	$(0,0,h)$
(M_F,q_0)	$(1,\ominus h,h)$	$(h,1\oplus h\circ\beta,0)$	$(h,0,1\ominus h)$	$(0,0,0)$
(M_G,q_0)	$(1\oplus h\circ\beta,0,\ominus h)$	$(0,1\ominus h,h)$	$(h,0,1)$	$(0,0,0)$
(M_H,q_0)	$(1,h,h)$	$(\ominus h,1\oplus h\circ\beta,0)$	$(h,0,1\ominus h)$	$(0,0,0)$
(M_I,q_0)	$(1,h,h)$	$(\ominus h\circ\beta,1,h)$	$(0,h,1\ominus h)$	$(0,0,0)$
(M_J,q_0)	$(1,0,\ominus h)$	$(0,1\ominus h\circ\beta,2\circ h)$	$(h\circ\beta,h,1)$	$(0,0,0)$
(M_K,q_0)	$(1,h,h)$	$(0,1\ominus h,0)$	$(\ominus h,0,1\oplus h\circ\beta)$	$(0,0,0)$
(M_L,q_0)	$(1,h\circ\beta_2,0)$	$(h,1\ominus h,h)$	$(h\circ\beta_1,0,1\oplus h)$	$(0,0,h)$
(M_M,q_0)	$(1,\ominus h,2\circ h\circ\beta)$	$(0,1\ominus h,h\circ\beta\oplus h)$	$(\ominus h,0,1\oplus h\circ\beta)$	$(0,0,h\circ\beta)$
(M_N,q_0)	$(1,h,h\ominus h\circ\beta)$	$(\ominus h\circ\beta,1,2\circ h)$	$(0,h,1\oplus h)$	$(0,0,h)$
(M_O,q_0)	$(1,h,h)$	$(h,1,h\circ\beta)$	$(0,\ominus h,1)$	$(0,0,0)$
(M_P,q_0)	$(1,\ominus h,h)$	$(h\circ\beta,1\oplus h,h)$	$(h,0,1)$	$(0,0,0)$
(M_Q,q_0)	$(1,h,h\circ\beta_1)$	$(0,1\ominus h,h\circ\beta_1\oplus h)$	$(\ominus h,0,1\oplus h\circ\beta_1\oplus h\circ\beta_2)$	$(0,0,h\circ\beta_1)$
(M_R,q_0)	$(1\oplus h\circ\beta_1,h\circ\beta_2,0)$	$(h\circ\beta_1\ominus h,1\oplus h\circ\beta_2,0)$	$(h\circ\beta_1,h\circ\beta_2\oplus h,1\ominus h)$	$(h\circ\beta_1,h\circ\beta_2,0)$
(M_S,q_0)	$(1\ominus h,h,2\circ h)$	$(\ominus h\circ\beta,1,h)$	$(0,h,1\oplus h)$	$(0,0,h)$

Рассмотрим инициальный автомат (M_x, q_0) ($q_0 = (0, 0, 1)$). Возможны следующие случаи.

Случай 1. $q_1 = (0, 0, q_1^{(3)})$. Пользуясь табл. 2, находим, что в этом случае $x \in \{A, B, C, E, L, R\}$. Установим автомат M_x в состояние $q_2 = (0, 0, 0)$. Тогда, согласно табл. 2, возможны 2 случая.

Случай 1.1. $q_3 = (0, 0, q_3^{(3)})$, $q_3^{(3)} \neq 0$. Тогда $x \in \{A, B, C, E, L\}$. Установим автомат M_x в состояние $q_4 = (\ominus 1, 1, 0)$. Подадим на автомат M_x входной символ 0. Тогда, согласно табл. 1, если $x = A$, то $q_5 = (\ominus 1\oplus h, 1\oplus h, 0)$; если $x = B$, то $q_5 = (\ominus 1, 1\ominus 2\circ h, 2\circ h)$; если $x = C$, то $q_5 = (\ominus 1, 1\ominus 2\circ h, 0)$; если $x = E$, то $q_5 = (\ominus 1, 1, h\circ(\beta\oplus 1))$; если $x = L$, то $q_5 = (\ominus 1\oplus h, h\circ(\beta_2\ominus 1), 2\circ h)$. Возможны 5 случаев.

Случай 1.1.1. $q_5^{(1)} \neq \ominus 1$, $q_5^{(3)} = 0$. Тогда $x = A$.

Случай 1.1.2. $q_5^{(1)} \neq \ominus 1$, $q_5^{(3)} \neq 0$. Тогда $x = L$.

Случай 1.1.3. $q_5^{(1)} = \ominus 1$, $q_5^{(2)} = 1$. Тогда $x = E$.

Случай 1.1.4. $q_5^{(1)} = \ominus 1$, $q_5^{(2)} \neq 1$, $q_5^{(3)} \neq 0$. Тогда $x = B$.

Случай 1.1.5. $q_5^{(1)} = \ominus 1$, $q_5^{(2)} \neq 1$, $q_5^{(3)} = 0$. Тогда $x = C$.

Случай 1.2. $q_3 = (q_3^{(1)}, q_3^{(2)}, 0)$. Тогда $x = R$.

Случай 2. $q_1 = (0, q_1^{(2)}, q_1^{(3)})$, $q_1^{(2)} \neq 0$. Пользуясь табл. 2, находим, что в этом случае $x \in \{D, I, J, N, O, R, S\}$.

Случай 2.1. $q_1^{(3)} = 1$. Тогда $x \in \{D, J, O\}$. Установим автомат M_x в состояние $q_2 = (1, 0, 0)$. Тогда, согласно табл. 2, возможны 3 случая.

Случай 2.1.1. $q_3 = (1, q_3^{(2)}, 0)$, $q_3^{(2)} \neq 0$. Тогда $x = D$.

Случай 2.1.2. $q_3 = (1, 0, q_3^{(3)})$, $q_3^{(3)} \neq 0$. Тогда $x = J$.

Случай 2.1.3. $q_3 = (1, q_3^{(2)}, q_3^{(3)})$, $q_3^{(2)} \neq 0$, $q_3^{(3)} \neq 0$. Тогда $x = D$.

Случай 2.2. $q_1^{(3)} \neq 1$. Тогда $x \in \{I, N, R, S\}$. Установим автомат M_x в состояние $q_2 = (0, 1, 0)$. Тогда, согласно табл. 2, возможны 2 случая.

Случай 2.2.1. $q_3 = (q_3^{(1)}, q_3^{(2)}, q_3^{(3)})$, $q_3^{(3)} \neq 0$. Тогда $x \in \{I, N, S\}$. Установим автомат M_x в состояние $q_4 = (1, 0, \ominus 1)$. Подадим на автомат M_x входной символ 0. Тогда, согласно табл. 1, если $x = I$, то $q_5 = (1, 0, \ominus 1 \oplus 2 \circ h)$; если $x = N$, то $q_5 = (1, 2 \circ h, \ominus 1 \oplus h \circ (1 \ominus \beta))$; если $x = S$, то $q_5 = (1 \ominus h, 2 \circ h, \ominus 1 \oplus 2 \circ h)$. Возможны 3 случая.

Случай 2.2.1.1. $q_5^{(1)} \neq 1$. Тогда $x = S$.

Случай 2.2.1.2. $q_5^{(1)} = 1$, $q_5^{(2)} = 0$. Тогда $x = I$.

Случай 2.2.1.3. $q_5^{(1)} = 1$, $q_5^{(2)} \neq 0$. Тогда $x = N$.

Случай 2.2.2. $q_3 = (q_3^{(1)}, q_3^{(2)}, 0)$. Тогда $x = R$.

Случай 3. $q_1 = (q_1^{(1)}, 0, q_1^{(3)})$, $q_1^{(1)} \neq 0$. Пользуясь табл. 2, находим, что в этом случае $x \in \{F, G, H, K, L, M, P, Q, R\}$.

Случай 3.1. $q_1^{(3)} \neq 1$. Тогда $x \in \{F, H, K, L, M, Q, R\}$.

Случай 3.1.1. $q_1^{(1)} \oplus q_1^{(3)} = 1$. Тогда $x \in \{F, H, K, L, M, Q, R\}$. Установим автомат M_x в состояние $q_2 = (1, 0, 0)$. Тогда, согласно табл. 2, возможны 3 случая.

Случай 3.1.1.1. $q_3^{(1)} \neq 1$. Тогда $x = R$.

Случай 3.1.1.2. $q_3^{(1)} = 1$, $q_3^{(3)} \neq 0$. Тогда $x \in \{F, H, K, M, Q\}$. Установим автомат M_x в состояние $q_4 = (0, 1, 0)$. Тогда, согласно табл. 2, возможны 3 случая.

Случай 3.1.1.2.1. $q_5^{(1)} \neq 0$, $q_3^{(2)} \neq q_3^{(3)}$. Тогда $x = F$.

Случай 3.1.1.2.2. $q_5^{(1)} \neq 0$, $q_3^{(2)} = q_3^{(3)}$. Тогда $x = H$.

Случай 3.1.1.2.3. $q_5^{(1)} = 0$. Тогда $x \in \{K, M, Q\}$. Установим автомат M_x в состояние $q_6 = (1, 1, 0)$. Подадим на автомат M_x входной символ 0. Тогда, согласно табл. 1, если $x = K$, то $q_7 = (1 \oplus h, 1, h)$; если $x = M$, то $q_7 = (1, 1 \ominus 2 \circ h, 3 \circ h)$; если $x = Q$, то $q_7 = (1, 1, h \circ (\beta_1 \oplus 1))$. Возможны 3 случая.

Случай 3.1.1.2.3.1. $q_7^{(1)} \neq 1$. Тогда $x = K$.

Случай 3.1.1.2.3.2. $q_7^{(1)} = 1$, $q_7^{(2)} \neq 1$. Тогда $x = M$.

Случай 3.1.1.2.3.3. $q_7^{(1)} = 1$, $q_7^{(2)} = 1$. Тогда $x = Q$.

Случай 3.1.1.3. $q_3^{(1)} = 1$, $q_3^{(3)} = 0$. Тогда $x \in \{L, Q\}$. Установим автомат M_x в состояние $q_4 = (1, 1, 0)$. Подадим на автомат M_x входной символ 0. Тогда, согласно табл. 1, если $x = L$, то $q_5 = (1 \oplus h, 1 \oplus h \circ (\beta_2 \ominus 1), 0)$; если $x = Q$, то $q_5 = (1, 1, h \circ (\beta_1 \oplus 1))$. Возможны 2 случая.

Случай 3.1.1.3.1. $q_5^{(1)} \neq 1$. Тогда $x = L$.

Случай 3.1.1.3.2. $q_5^{(1)} = 1$. Тогда $x = Q$.

Случай 3.1.2. $q_1^{(1)} \oplus q_1^{(3)} \neq 1$. Тогда $x \in \{K, L, M, Q, R\}$. Установим автомат M_x в состояние $q_2 = (0, 1, 0)$. Тогда, согласно табл. 2, возможны 3 случая.

Случай 3.1.2.1. $q_3^{(1)} = 0$. Тогда $x \in \{K, M, Q\}$. Этот случай аналогичен случаю 3.1.1.2.3.

Случай 3.1.2.2. $q_3^{(1)} \neq 0$, $q_3^{(3)} \neq 0$. Тогда $x = L$.

Случай 3.1.2.3. $q_3^{(1)} \neq 0$, $q_3^{(3)} = 0$. Тогда $x = R$.

Случай 3.2. $q_1^{(3)} = 1$. Тогда $x \in \{G, K, M, P, Q\}$. Установим автомат M_x в состояние $q_2 = (0, \ominus 1, 0)$. Подадим на автомат M_x входной символ 0. Тогда $h = q_3^{(2)} \oplus 1$ для всех M_x ($x \in \{G, K, M, P, Q\}$). Установим автомат M_x в состояние $q_4 = (1, 0, 1)$. Подадим на автомат M_x входной символ 0. Тогда, согласно табл. 1, если $x = G$, то $q_5 = (1 \oplus h \circ (\beta \oplus 1), h, 1 \ominus h)$; если $x = K$, то $q_5 = (1 \ominus h, h, 1 \oplus h)$; если $x = M$, то $q_5 = (1 \ominus h, \ominus h, 1)$; если $x = P$, то $q_5 = (1 \ominus h, h, 1)$. Возможны 5 случаев.

Случай 3.2.1. $q_5^{(3)} = 1 \ominus h$. Тогда $x = G$.

Случай 3.2.2. $q_5^{(3)} = 1$, $q_5^{(2)} = \ominus h$. Тогда $x = M$.

Случай 3.2.3. $q_5^{(3)} = 1$, $q_5^{(2)} = h$. Тогда $x = Q$.

Случай 3.2.4. $q_5^{(3)} = 1 \oplus h$, $q_5^{(2)} = h$. Тогда $x = K$.

Случай 3.2.5. $q_5^{(3)} = 1 \oplus h$, $q_5^{(2)} = \ominus h$. Тогда $x = P$.

Случай 4. $q_1 = (q_1^{(1)}, q_1^{(2)}, q_1^{(3)})$, $q_1^{(1)} \neq 0$, $q_1^{(2)} \neq 0$. Пользуясь табл. 2, находим, что в этом случае $x \in \{J, R\}$.

Случай 4.1. $q_1^{(3)} = 1$. Тогда $x = J$.

Случай 4.2. $q_1^{(3)} \neq 1$. Тогда $x = R$.

Таким образом, каждый автомат M_x ($x \in Y$) можно идентифицировать n-кратным экспериментом высоты 1, где $n \leq 4$.

В случае, когда информационная переменная добавляется только в некоторые уравнения системы для автомата Спротта, задача значительно усложняется за счет того, что криптоаналитику становится недоступной для непосредственного наблюдения по крайней мере одна из фазовых переменных $q_t^{(i)}$ ($i = 1, 2, 3$).

Заключение

В работе решена задача идентификации автомата в подклассе автоматов Спротта, для которых информационная переменная добавлена во все уравнения, определяющие автоматы. Показано, что в этом случае криптоаналитик сможет идентифицировать автомат, проделав 4-кратный эксперимент высоты 1.

Задача идентификации автомата была решена в предположении, что экспериментатор может управлять входом автомата и осуществлять инициализацию автомата требуемое число раз. Ясно, что при том или ином ограничении этих возможностей сложность решения задачи возрастет. Детальное исследование роста такой сложности представляет собой одно из возможных направлений дальнейших исследований.

Актуальность задачи обусловлена применением исследуемых автоматов в качестве шифрсистем, такое исследование дает возможность теоретически и с единых позиций оценить стойкость поточного шифра, определяемого автоматом Спротта при использовании информационного потока в качестве управления.

ЛИТЕРАТУРА

1. *Сухинин В.А.*, *Скобелев В.Г.* Эквивалентность состояний систем Спротта // Труды ИПММ НАН Украины. 2007. Т. 14. С. 174 – 186.
2. *Сухинин В.А.*, *Скобелев В.Г.* Алгоритмы и сложность идентификации автоматов Спротта над кольцом $\mathbf{Z}_p^k$ // Труды VII Междунар. конф. «Идентификация систем и задачи управления» SICPRO'08. Москва, 28 – 31 января 2008 г. ИПУ РАН. С. 1107 – 1153.
3. *Кузнецов С.П.* Динамический хаос. М.: Физматлит, 2001. 296 с.

СВЕДЕНИЯ ОБ АВТОРАХ

АБРОСИМОВ Михаил Борисович – кандидат физико-математических наук, доцент кафедры теоретических основ компьютерной безопасности и криптографии Саратовского государственного университета. E-mail: **mic@rambler.ru**

АГИБАЛОВ Геннадий Петрович – профессор, доктор технических наук, заведующий кафедрой защиты информации и криптографии Томского государственного университета. E-mail: **agibalov@isc.tsu.ru**

БОРОВКОВ Сергей Иванович – студент Томского государственного университета систем управления и радиоэлектроники.

БУШКОВ Виктор Георгиевич – студент радиофизического факультета Томского государственного университета. E-mail: **bushkov@sibmail.com**

ГОРШКОВ Сергей Павлович – старший научный сотрудник, кандидат физико-математических наук, доцент Института криптографии, связи и информатики Академии ФСБ России, г. Москва. E-mail: **spg54@bk.ru**

ДЕВЯНИН Петр Николаевич – доцент, доктор технических наук, заместитель заведующего кафедрой программирования и компьютерной безопасности Института криптографии, связи и информатики Академии ФСБ России, г. Москва. E-mail: **peter_devyanin@hotmail.com**

ДОЛГОВ Александр Алексеевич – студент Саратовского государственного университета.

ЕВДОКИМОВ Александр Андреевич – профессор, кандидат физико-математических наук, заведующий лабораторией Института математики им. С.Л. Соболева СО РАН, г. Новосибирск. E-mail: **evdok@math.nsc.ru**

ЕВСЮТИН Олег Олегович – студент Томского государственного университета систем управления и радиоэлектроники.

ИВАНОВ Алексей Владимирович – аспирант Московского государственного института радиотехники, электроники и автоматики. E-mail: **alexiva@pochta.ru**

КИРКО Ирина Николаевна – профессор кафедры информационной безопасности Сибирского федерального университета, зам. директора Регионального учебно-научного центра по проблемам информационной безопасности в системе высшей школы (РУНЦ ИБ), г. Красноярск. E-mail: **kin@fivt.krgtu.ru**

КОЛЕГОВ Денис Николаевич – аспирант Томского государственного университета. E-mail: **kden@sibmail.com**

КУТЬИН Александр Михайлович – кандидат физико-математических наук, доцент кафедры СИИ Сибирского федерального университета, г. Красноярск. E-mail: **maglinetc@mail.ru**

КУЧЕРОВ Михаил Михайлович – профессор кафедры информационной безопасности Сибирского федерального университета, г. Красноярск. E-mail: **kucherov@fivt.krgtu.ru**

КЫРОВ Владимир Александрович – доцент Горно-Алтайского государственного университета. E-mail: **kfizika@ gasu.ru**

ЛЕВИН Альберт Абрамович – кандидат физико-математических наук, старший научный сотрудник Института математики им. С.Л. Соболева СО РАН, г. Новосибирск. E-mail: **levin@math.nsc.ru**

МУЛЛЕР Алексей Аронович – аспирант кафедры информационной безопасности Сибирского федерального университета, г. Красноярск.

НАУМЕНКО Иван Александрович – студент Донецкого национального технического университета. E-mail: **e-one@bk.ru**

ПУДОВКИНА Марина Александровна – кандидат физико-математических наук, доцент кафедры криптологии и дискретной математики Московского инженерно-физического института. E-mail: **maricap@rambler.ru**

РОСОШЕК Семён Константинович – кандидат физико-математических наук, доцент кафедры алгебры Томского государственного университета. E-mail: **rososhek@list.ru**

САЛИЙ Вячеслав Николаевич – профессор, кандидат физико-математических наук, заведующий кафедрой теоретических основ компьютерной безопасности и криптографии Саратовского государственного университета. E-mail: **SaliiVN@info.sgu.ru**

СКОБЕЛЕВ Владимир Владимирович – аспирант Института прикладной математики и механики НАН Украины, г. Донецк. E-mail: **skobelev_vladimir@hotmail.com**

СКОБЕЛЕВ Владимир Геннадиевич – профессор, доктор технических наук, ведущий научный сотрудник Института прикладной математики и механики НАН Украины, г. Донецк. E-mail: **skbv@iamm.ac.donetsk.ua**

СТЕФАНЦОВ Дмитрий Александрович – студент Томского государственного университета. E-mail: **d.a.stephantsov @gmail.com**

СУХИНИН Владимир Александрович – аспирант Донецкого национального университета. E-mail: **vladimir.suhinin @gmail.com**

ТОКАРЕВА Наталья Николаевна – аспирантка Института математики им. С.Л. Соболева СО РАН, г. Новосибирск. E-mail: **tokareva@math.nsc.ru**

ЧЕРЕМУШКИН Александр Васильевич – доктор физико-математических наук, заведующий кафедрой программирования и компьютерной безопасности Института криптографии, связи и информатики Академии ФСБ России, г. Москва. E-mail: **avc238@mail.ru**

ШОЛОМОВ Лев Абрамович – профессор, доктор физико-математических наук, ведущий научный сотрудник Института системного анализа РАН, г. Москва. E-mail: **sholomov@isa.ru**

АННОТАЦИИ СТАТЕЙ НА АНГЛИЙСКОМ ЯЗЫКЕ

Abrosimov M.B., *Dolgov A.A.* **FAMILIES OF TOURNAMENT'S EXTENSIONS.** Graph $G^* = (V^*, \alpha^*)$ is said to be an *exact k-extension* of a graph $G = (V, \alpha)$ if every graph obtained by removing any k vertexes from G^* and graph G are isomorphic. We present some basic conditions that directed graph must meet to be an exact k-extension. We then study the problem of constructing exact k-extension of tournaments. Presented four families of tournaments with their exact extensions. We show some experimental data collected during study exact extensions of tournaments.

Agibalov G.P. **SOME THEORETICAL ASPECTS OF DIFFERENTIAL CRYPTANALYSIS OF THE ITERATED BLOCK CIPHERS WITH ADDITIVE ROUND KEY.** A common method is formulated for differential cryptanalysis of any iterated block cipher with additive round key. For a r-round cipher, the method uses its $(r–1)$-round characteristic and an algorithm for solving the system of polynomial equations over a finite field describing for the rth round the dependencies of its known outputs on its input differences known with a nonzero probability and on its unknown inputs and round key elements. Before the method is stated necessary notions are introduced and basic propositions are given concerning the functions defined on finite abelian groups, iterated block ciphers, their round functions, additive round keys, differential characteristics, round system of equations over a finite field, and so on. All the notions and assertions are illustrated on DES.

Bushkov V.G. D**ESCRIBING PROGRESSIVE SOLUTIONS TO A PARALLEL FSM EQUATION.** We consider the problem of deriving a component X that combined with the known part of system C, called the context, conforms to a given overall specification S. Many problems of logic synthesis and analysis can be reduced to solving a parallel Finite State Machine (FSM) equation $C \lozenge X = S$. However, not each solution to the equation is of practical use. In this paper, we are interested in progressive solutions, i.e. solutions which have no livelocks without exit when composed with the context. We propose a novel algorithm for deriving a largest progressive solution by removing non-progressive strings from the largest solution to the equation. Since, in general, the number of such strings is infinite, we show that a proposed algorithm terminates.

Cheremushkin A.V. **COMBINATORIAL – GEOMETRIC TECHNIQUE TO DESIGN A KEY DISTRIBUTION PATTERNS (AN OVERVIEW).** The paper presents an overview of known approaches to design a key distribution patterns and their applications to the problem of construction key predistribution schemes.

Devyanin P.N. **BASE ROLE DP-MODEL.** This article represents a base role DP-model constructed on the basis of group role-based access control (RBAC) models and DP-models computer systems with discretionary or mandatory access control. The singularities of functioning modern computer systems take into account in base role DP-model including distinctions in conditions of functioning and the cooperation of trusted and nontrusted user sessions, in conditions of realization of information flows by memory or by time, and also possibility of getting the nontrusted session control above the trusted session while realization of information flows by memory on entity which is functionally associated with the trusted session. Monotonous and nonmonotonic rules of transformation of conditions of system are described in details and analyzed while sets of actual roles, rights of access and possible actions of the nontrusted sessions are used. Sufficient conditions of access rights transfer roles are substantiated by user sessions.

Evdokimov A.A., *Levin A.A.* **A TOOLKIT FOR GRAPHICAL ANALYSIS OF WORD CHAINS.** The authors have developed a software package "BruijnViz" intended for researching properties of unlimited word chains. All the subwords of length n taken from a symbol sequence are depicted on de Bruijn graph. The graph of subwords is named the graph-portrait of the sequence. The set of the graph-portraits of a word chain constructed for $n = 1, 2, 3, ...$ characterizes the chain: its periodicity, length and number of repetitions of subwords, variety of subwords, their structure, etc. The package represents each graph-portrait of a word chain during its growth. The graph-portraits of many known mathematical chains and of the word chains originated in different applications are presented in the paper.

Gorshkov S.P. **ON THE COMPLEXITY OF WEAKLY POSITIVE AND WEAKLY NEGATIVE BOOLEAN FUNCTIONS REDUCING.** Boolean function $f(x_1, ..., x_k)$ is called *weakly positive* (*weakly negative*) if f may be represented by CNF such as $f \equiv \bigwedge_{i=1}^{t} (x_{s_{i1}}^{\alpha_i} \vee x_{s_{i2}} \vee ... \vee x_{s_{ik_i}})$, where $\alpha_i \in \{0, 1\}$, $i = 1, ..., t$, (respectively by CNF such as $f \equiv \bigwedge_{i=1}^{t} (x_{s_{i1}}^{\alpha_i} \vee \overline{x}_{s_{i2}} \vee ... \vee \overline{x}_{s_{ik_i}})$, where $\alpha_i \in \{0, 1\}$, $i = 1, ..., t$). These formulas are called reduced representations of weakly positive and weakly negative functions accordingly. The complexity of reducing the weakly positive and weakly negative functions represented by perfect CNF or algebraic normal form is evaluated in this paper.

Ivanov A.V. **APPROXIMATION OF PLATEAUED BOOLEAN FUNCTIONS BY MONOMIAL ONES.** From the Parseval's equation we have that if the squared Walsh transform of the Boolean function takes at most one nonzero value then its Walsh coefficients are equal to 2^{2n-2s} for some $s \le n/2$. These functions are called the $2s$ – order plateaued functions. In the present paper we consider the aspects of an approximation of the plateaued functions by monomial ones. We use the representation of n-variable Boolean functions by polynomials over the field $\mathbb{F}_{2^n}$. The necessary conditions for the Boolean functions to have the

Hamming distance to all bijective monomials taking only three values: 2^{n-1}, $2^{n-1} \pm 2^{n-s-1}$, are obtained. The non-existence of the functions, satisfying these conditions for such n that $2^n - 1$ is prime, is shown.

Kirov V.A. **THREE-BASAL QUASIGROUP WITH GENERALIZED WORD'S IDENTITY.** This article touches upon three-basal partial quasigroup with generalized Word's identity. This quasigroup is proved to be phenomenologically symmetric. Phenomenologically symmetric subquasigroups are studied.

Kolegov D.N. **DP-MODEL APPLICATION FOR NETWORK SECURITY ANALYSIS.** The closure of the DP-models is offered. The algorithms of finding closure are described. Application of the models for network security analyses is considered. REM and VTG topological models are formalized in terms of DP-model theory.

Kutcherov M.M., *Kirko I.N.*, *Muller A.A.* **MODERN MODELS AND MECHANISMS IN INFORMATION SECURITY.** Increasingly, organizations are developing sophisticated computing systems on whose services they need to place great trust. In many circumstances the focus will be made on different properties of such services – e.g., on the integrity, on the average real-time response achieved, or on the degree to which deliberate intrusions can be prevented. Simultaneous consideration of availability and integrity provides a very convenient means of subsuming these various concerns within a single conceptual framework. It also provides the means of addressing the problem of an appropriate balance of these properties. The Lattice model, Chinese Wall policy, and Grosch's law are central to the understanding and mastering of our model. The model provided for achieving availability and integrity is extremely useful, as before those attributes have been considered much more orthogonal to each other, and the development of any real system should have performed trade offs. The model enables the more classical notions of integrity, availability, and confidentiality to be put into perspective.

Kut'in A.M. **CODES**, **COMPOSITIONS AND LATTICES**. The questions of the coding theory, connected with its central problem are discussed. The principle model of the coding theory and cryptography – the distributive lattice – is considered. Its connection with the number composition and c-matroid is investigated. The lattice codes are created. The limits of the codes in the Hamming's metric are determined.

Naumenko I.A., *Skobelev V.G.* **TRANSFORMATION OF GRIDS TO 1-FAULT-TOLERANT GRAPHS.** For some n-dimensional grids it is investigated the problem of their transformation to 1-fault-tolerant. The method of transformation is based on translations determined by cyclic group. Some characteristics of resulted graphs are estimated.

Pudovkina M.A. **LINEAR STRUCTURES OF PERMUTATION GROUPS OVER FINITE MODULES**. It is known that mappings with linear structures are cryptographically bad. In this paper we describe biective mappings over finite modules with linear structures using permutation groups.

Rososhek S.K., *Borovkov A.A.*, *Evsutin O.O.* **CELLULAR AUTOMATON CRYPTOSYSTEMS.** The new class of cryptosystems based on reversible cellular automatons is constructed. Full description of two examples of cellular automaton cryptosystems is given: namely cryptosystem based on cellular automaton with Moore neighbor and cryptosystem based on cellular automaton on partitioning. First type cryptosystem uses key for controlling over process of cellular automaton rule change by which every cell computes next state from his neighbors current states. Second type cryptosystem uses key for controlling over cellular automaton on portioning dynamics development, namely for controlling over following parameters: lattice block height, lattice block width, shift size in rule table and step number of cellular automaton development.

Salii V.N. **MINIMAL PRIMITIVE EXTENSIONS OF ORIENTED GRAPHS**. (Oriented) graph $G = (V, \alpha)$ is called primitive if there exists an integer $r \geq 1$ such that every two vertices can be connected by a route of length r. A graph $G' = (V, \alpha)$ is said to be a primitive extension of G if G' is primitive and $\alpha \subseteq \alpha'$. Primitive extensions with a minimal possible number of additional arcs are constructed for some acyclic graphs (trees, linear and polygonal graphs).

Sholomov L.A. **ENTROPY OF UNDERDETERMINED SEQUENCES UNDER CONSTRAINTS TO SPECIFICATIONS.** Sequences of undetermined symbols are considered. Every such symbol is associated with a set of completely defined symbols, one of which can be used to replace (specify) the former. The specification of an undetermined sequence is the result of replacing all of its symbols by some specifications. We consider the classes of all undetermined sequences which have given length and given frequencies of symbols appearance. The combinatorial entropy of the class is logarithm of the minimum cardinality of a set which contains a specification for each sequence of the class. If some constraints to parameters of specifications are introduced we say about the constrained entropy. The article contains the fairly sharp estimates of the constrained combinatorial entropy.

Skobelev V.V. **CHARACTERISTICS OF FIXED POINTS FOR LINEAR AUTOMATA OVER A FINITE RING.** We investigate the structure of the set of fixed points for mappings that are implemented via initial linear Mealy and Moore automata over the ring $\mathbf{Z}_{p^k}$. Input symbols that are fixed points for current state of investigated automata are also characterized.

Stephantsov D.A. **IMPLEMENTATION OF SECURITY POLICIES IN COMPUTER SYSTEMS BY MEANS OF ASPECT-ORIENTED PROGRAMMING.** Aspect-Oriented Programming is considered as the mean to implement security policies. Short description of AspectJ programming language is given along with security aspect implementation example. The notions of Object-Oriented Programming and Meta-Object Protocol are introduced showing the way of implementing Aspect-Oriented Programming by means of Meta-Object Protocol. AspectTalk programming language which was implemented by author is briefly introduced. Simple security policy implementation in AspectTalk is given and commented.

Sukhinin V.A., *Skobelev V.G.* **AUTOMATON IDENTIFICATION IN THE CLASS OF SPROTT'S AUTOMATA.** The problem of automaton identification in the class of Sprott's automata over the finite ring $\mathbf{Z}_p^k = (\mathbf{Z}_p^k, \oplus, \circ)$ is solved. The estimation problem of the solution complexity is considered as a necessary characteristics for computational security of a stream cipher determined by the Sprott's automaton when information stream is interpreted as some control. It is shown that in a Sprott's automata subclass the problem is solved by carrying out multiple experiments with automaton.

Tokareva N.N. **QUADRATIC APPROXIMATIONS OF THE SPECIAL TYPE FOR THE 4-BIT PERMUTATIONS IN S-BOXES.** We consider quadratic approximations (for Boolean functions) of the special type and study the ability of application them in nonlinear cryptanalysis of block ciphers. For 4-bit permutations with the most high nonlinearity recommended for using in S-boxes of GOST 28147-89, DES, s^3DES we show that for the all of them (excepting only one) there are quadratic equalities of the special type on input and output bits with probability more high than any linear equality has.