ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

ПРИКЛАДНАЯ ДИСКРЕТНАЯ МАТЕМАТИКА

Научный журнал

2009 **№1(3)**

Свидетельство о регистрации: ПИ №ФС 77-33762
от 16 октября 2008 г.

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 её приложений в криптографии, компьютерной безопасности, кибернетике, информатике, программировании, теории надежности, интеллектуальных системах.

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. А. Стефанцов*

Изд. лиц. ИД. №04000 от 12.02.2001. Подписано к печати 16.03.2009
Формат $60 \times 84\frac{1}{8}$. Бумага офсетная. Печать офсетная. Гарнитура «Таймс».
Усл. п. л. 14, 88. Уч.-изд. л. 16, 66. Тираж 300 экз. Заказ №8.

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

СОДЕРЖАНИЕ

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ В ДИСКРЕТНОЙ МАТЕМАТИКЕ

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КРИПТОГРАФИИ

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

CONTENTS

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПРИКЛАДНОЙ ДИСКРЕТНОЙ МАТЕМАТИКИ

DOI 10.17223/20710410/3/1

УДК 519.7

МИНИМИЗАЦИЯ БУЛЕВЫХ ФУНКЦИЙ МНОГИХ ПЕРЕМЕННЫХ В КЛАССЕ ДНФ — ИТЕРАТИВНЫЙ МЕТОД И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

А. Д. Закревский, Н. Р. Торопов

Объединенный институт проблем информатики НАН Беларуси, г. Минск, Беларусь

E-mail: zakrevskij@tut.by

Предлагается итеративный алгоритм минимизации булевых функций многих переменных, основанный на использовании параллельных операций над соседними элементами в булевом пространстве аргументов. Он включает операцию быстрого нахождения элементов характеристического множества с малым числом соседей и формирования определяемых ими импликант, итеративную процедуру применения этой операции к последовательно сокращаемому характеристическому множеству и операцию приведения множества полученных конъюнктов к корректной ДНФ.

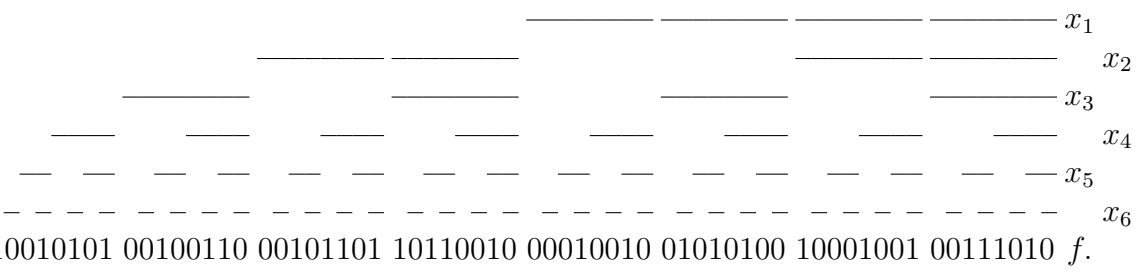
Ключевые слова: *булевы функции, ДНФ, минимизация.*

Введение

Классическая задача минимизации булевых функций в классе дизъюнктивных нормальных форм (ДНФ) может быть сформулирована следующим образом.

Определим произвольную булеву функцию $f(\boldsymbol{x}) \equiv f(x_1, x_2, \ldots, x_n)$ стандартным образом булевым вектором $\boldsymbol{f}$ с 2^n компонентами, пронумерованными слева направо, начиная с нуля, причем компонента с номером k задает значение функции f на наборе значений аргументов, представляющем общепринятый двоичный код $\boldsymbol{b}^k$ числа k.

Например, следующий вектор $\boldsymbol{f}$ представляет булеву функцию f шести переменных x_1, x_2, x_3, x_4, x_5, x_6, принимающую значение 1 на 27 наборах значений аргументов, образующих характеристическое множество $M^1 = \{000000, 000011, 000101, \ldots, 111110\}$ функции f. Порядковые номера соответствующих компонент вектора $\boldsymbol{f}$ равны 0, 3, 5, …, 62.



Заметим, что такой вектор можно интерпретировать как совершенную нормальную форму булевой функции, члены которой представлены единичными компонентами вектора и их кодами. Например, код $\boldsymbol{b}^{10} = 001010$ элемента f_{10} задает полную элементарную конъюнкцию $\overline{x}_1\overline{x}_2x_3\overline{x}_4x_5\overline{x}_6$.

Проблема заключается в том, чтобы представить заданную вектором $\boldsymbol{f}$ функцию f в ДНФ, желательно минимизированной. Заметим, что традиционные методы решения этой задачи связаны с затратами времени, быстро растущими с ростом n, и становятся практически неприемлемы при $n > 20$, когда число членов в совершенной ДНФ исчисляется миллионами [1]. В связи с этим в данной работе предлагается оригинальный метод получения ДНФ булевых функций, число аргументов которых может достигать 25. Метод основан на применении эффективных параллельных операций над булевыми 2^n-векторами, предложенных в работах [2, 3].

К таким операциям относится, в частности, операция конъюнктивного симметрирования вектора $\boldsymbol{f}$ по переменной x_i, обозначаемая ниже через $\mathrm{S}\,\boldsymbol{f} \wedge i$. При ее выполнении вектор $\boldsymbol{f}$ разбивается на 2^{n-1} пар компонент, соседних по переменной x_i, и оба элемента каждой пары получают значение, равное конъюнкции исходных значений этих элементов. Напомним, что *соседними* называются такие компоненты вектора f, которым соответствуют наборы значений аргументов, различающиеся ровно в одном аргументе.

Заметим,что при числе переменных, превышающем 5, удобно представлять вектор $\boldsymbol{f}$ в виде булевой матрицы размером 2^5 на 2^{n-5}, отображая её 32-компонентные строки словами в компьютерной памяти (что адекватно для большинства современных компьютеров). Тогда любые два элемента вектора $\boldsymbol{f}$, соседние по переменной x_i, будут принадлежать к одному слову, если $i < 6$, и к разным словам в противном случае, что можно использовать для ускорения вычислений.

1. Построение булевой матрицы соседства $\boldsymbol{N}$

На первом этапе предлагаемого метода посредством n-кратного применения операции $\mathrm{S}\,\boldsymbol{f} \wedge i$ строится булева *матрица соседства* N размером $n \times 2^n$. Каждая строка $\boldsymbol{n}_i$ этой матрицы представляет результат выполнения операции $\mathrm{S}\,\boldsymbol{f} \wedge i$ при конкретном значении параметра i (от 1 до n). Матрица N отображает структуру характеристического множества M^1 функции $f(\boldsymbol{x})$, на котором эта функция принимает значение 1. Элемент n_i^k матрицы N принимает значение 1, если и только если k-й элемент вектора $\boldsymbol{f}$ равен 1 и имеет соседа по переменной x_i, также со значением 1. Таким образом, i-я строка этой матрицы отображает (единицами) подмножество элементов из M^1, имеющих в этом же множестве соседей по переменной x_i, а k-й столбец показывает, по каким переменным имеет соседей соответствующий элемент из M^1, представленный k-й компонентой вектора $\boldsymbol{f}$.

Продемонстрируем получение матрицы $\boldsymbol{N}$ на примере того же вектора $\boldsymbol{f}$, задающего случайную булеву функцию шести переменных:

$$f = 10010101\ 00100110\ 00101101\ 10110010\ 00010010\ 01010100\ 10001001\ 00111010,$$

$$N = \begin{array}{l} 00010000\ 00000100\ 00001001\ 00110010\ 00010000\ 00000100\ 00001001\ 00110010 \\ 00000101\ 00100010\ 00000101\ 00100010\ 00000000\ 00010000\ 00000000\ 00010000 \\ 00000100\ 00000100\ 00100000\ 00100000\ 00010000\ 00010000\ 00001000\ 00001000 \\ 00010001\ 00100010\ 00000000\ 00100010\ 00000000\ 01000100\ 10001000\ 00100010 \\ 00000101\ 00000000\ 00000101\ 10100000\ 00000000\ 01010000\ 00000000\ 00001010 \\ 00000000\ 00000000\ 00001100\ 00110000\ 00000000\ 00000000\ 00000000\ 00110000. \end{array}$$

2. Матричное представление ДНФ

В аналогичной форме булевыми вектором $\boldsymbol{g}$ и матрицей $\boldsymbol{D}$ той же размерности, называемыми *вектором решения* и *матрицей решения*, предлагается представлять и искомую ДНФ, рассматриваемую как некоторую совокупность интервалов булева пространства $M = \{0,1\}^n$. В векторе $\boldsymbol{g}$ отмечаются некоторые содержащиеся в интервалах элементы множества M^1, по одному для каждого интервала, а в столбцах матрицы $\boldsymbol{D}$ отмечаются внутренние переменные этих интервалов.

Рассматривая векторы $\boldsymbol{f}$ и $\boldsymbol{g}$ как множества отмеченных в них элементов пространства M, а матрицы $\boldsymbol{N}$ и $\boldsymbol{D}$ — как подмножества декартова произведения множеств $\boldsymbol{x}$ и M, сформулируем очевидное

Утверждение 1. $\boldsymbol{g} \subseteq \boldsymbol{f}$ и $\boldsymbol{D} \subseteq \boldsymbol{N}$.

Другими словами, вектор решения $\boldsymbol{g}$ и матрица решения $\boldsymbol{D}$ могут быть получены соответственно из вектора $\boldsymbol{f}$ и матрицы $\boldsymbol{N}$ заменой в них некоторых единиц на нули, как это и делается в описываемом в данной статье методе.

Для рассматриваемого примера искомая ДНФ будет выглядеть следующим образом:

$$g = 10010100\ 00000110\ 00101000\ 10010000\ 00000010\ 01000000\ 10000001\ 00001000,$$

$$D = \begin{matrix} 00010000\ 00000100\ 00000000\ 00010000\ 00000000\ 00000000\ 00000001\ 00000000 \\ 00000100\ 00000010\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000 \\ 00000000\ 00000000\ 00100000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000 \\ 00000000\ 00000010\ 00000000\ 00000000\ 00000000\ 00000000\ 10000000\ 00000000 \\ 00000100\ 00000000\ 00000000\ 10000000\ 00000000\ 01000000\ 00000000\ 00001000 \\ 00000000\ 00000000\ 00001000\ 00010000\ 00000000\ 00000000\ 00000000\ 00000000. \end{matrix}$$

В более известной форме эта ДНФ задается троичной матрицей, столбцы которой представляют элементарные конъюнкции ($\overline{x}_1\overline{x}_2\overline{x}_3\overline{x}_4\overline{x}_5\overline{x}_6, \overline{x}_2\overline{x}_3\overline{x}_4x_5x_6, \ldots$ и т. д.)

$$\begin{matrix} 1\ 0-0-0\,0\,0\,0-1\,1\ 1-1 \\ 2\ 0\,0-0-1\,1\,1\,1\,0\,0\ 1\,1\,1 \\ 3\ 0\,0\,0\,1\,1-0\ 1\,1\,0\,1\,0\,0\,1 \\ 4\ 0\,0\,1\,1-0\ 1\,0\,0\,1\,0-1\,1 \\ 5\ 0\,1-0\ 1\,1\,0-1\,1-0\ 1- \\ 6\ 0\,1\,1\,1\,0\,0-0-0\,1\,0\ 1\,0. \end{matrix}$$

Число p конъюнкций в полученной ДНФ равно весу вектора решения $\boldsymbol{g}$, а длина ДНФ (сумма рангов конъюнкций) равна $pn - q$, где q — число единиц в матрице $\boldsymbol{D}$.

Заметим, что введенные выше булевы матрицы и векторы при программной реализации предлагаемого метода обрабатываются во внутренних циклах и, следовательно, должны быть представлены в оперативной памяти, что ограничивает их допустимый размер. Учитывая параметры персонального компьютера, можно утверждать, что данный метод достаточно быстро реализуется на нем, если число переменных минимизируемой булевой функции не превышает 25.

3. Выделение элементов с малым числом соседей

Построение реализующей функцию f ДНФ целесообразно начать с поиска элементов ядра решения — обязательных простых импликант высокого ранга. Назовем обязательной такую простую импликанту функции f, которая входит в любую кратчайшую

ДНФ этой функции. Поиск облегчается предварительным выделением в векторе $\boldsymbol{f}$ элементов характеристического множества M^1 с небольшим числом соседей, поскольку именно такие элементы могут определять импликанты высокого ранга, отображаемые элементарными конъюнкциями со многими литералами.

Число соседей у элементов вектора $\boldsymbol{f}$ со значением 1 можно представить конечнозначным вектором $\boldsymbol{w}$

$$\begin{array}{l} f = 10010101\ 00100110\ 00101101\ 10110010\ 00010010\ 01010100\ 10001001\ 00111010 \\ w = 0..2.3.3\ \ ..2..22.\ \ ..1.23.3\ 1.62..3.\ \ ...2..0.\ \ .2.3.2..\ 1...3..1\ \ ..332.3., \end{array}$$

однако более удобно, в перспективе программной реализации последующих вычислений, рассортировать элементы по числу соседей и представить результат серией булевых векторов $\boldsymbol{m}_i$, в которых единицами отмечены элементы с i соседями.

Для рассматриваемого примера, при $i < 4$, получим

$$\begin{array}{rl} f = & 10010101\ 00100110\ 00101101\ 10110010\ 00010010\ 01010100\ 10001001\ 00111010, \\ \boldsymbol{m}_0 = & 10000000\ 00000000\ 00000000\ 00000000\ 00000010\ 00000000\ 00000000\ 00000000, \\ \boldsymbol{m}_1 = & 00000000\ 00000000\ 00100000\ 10000000\ 00000000\ 00000000\ 10000001\ 00000000, \\ \boldsymbol{m}_2 = & 00010000\ 00100110\ 00001000\ 00010000\ 00010000\ 01000100\ 00000000\ 00001000, \\ \boldsymbol{m}_3 = & 00000101\ 00000000\ 00000101\ 00000010\ 00000000\ 00010000\ 00001000\ 00110010. \end{array}$$

Эти векторы, образующие соответствующую булеву матрицу $\boldsymbol{M}$, легко получить эффективными покомпонентными операциями над строками матрицы $\boldsymbol{N}$, что существенно ускоряет поиск подходящих импликант.

4. Нахождение простых импликант

Обозначим через $\boldsymbol{t}^k$ троичный вектор, получаемый из вектора $\boldsymbol{b}^k$ (кода элемента f_i) присвоением значения $-$ компонентам, отмеченным единицами в столбце $\boldsymbol{n}^k$ матрицы $\boldsymbol{N}$. Его можно интерпретировать как некоторый интервал Int_k пространства $M = \{0,1\}^n$, а также как соответствующую элементарную конъюнкцию, которая может оказаться простой импликантой функции f.

Утверждение 2. Вектор $\boldsymbol{t}^k$ представляет обязательную простую импликанту функции f, если и только если $\mathrm{Int}_k \subseteq M^1$.

Утверждение 3. Для каждой обязательной простой импликанты функции f в матрице $\boldsymbol{N}$ найдется столбец $\boldsymbol{n}^k$, соответствующий которому троичный вектор $\boldsymbol{t}^k$ представляет эту импликанту.

Легко находятся обязательные простые импликанты ранга n — они непосредственно представляются элементами множества M^1, не имеющими соседей. Они отмечаются в векторе $\boldsymbol{m}_0$ и представляются соответствующими столбцами матрицы $\boldsymbol{N}$, не содержащими единиц. Аналогично, все обязательные простые импликанты ранга $n-1$ отмечены в векторе $\boldsymbol{m}_1$ и также представлены соответствующими столбцами матрицы $\boldsymbol{N}$ — они содержат по одной единице.

Выявление обязательных простых импликант меньшего ранга несколько сложнее. Однако оно достаточно просто для ранга $n-2$ и $n-3$, осуществляясь посредством покомпонентных операций над некоторыми столбцами матрицы соседства $\boldsymbol{N}$.

Утверждение 4. Предположим, что k-й элемент вектора $\boldsymbol{f}$ имеет двух соседей. Тогда вектор $\boldsymbol{t}^k$ представляет обязательную простую импликанту функции f, если и только если j-я компонента вектора $\boldsymbol{f}$ равна единице, где $\boldsymbol{b}^j = \boldsymbol{b}^k \oplus \boldsymbol{n}^k$.

Например, $\boldsymbol{b}^{27} \oplus \boldsymbol{n}^{27} = 011011 \oplus 100001 = 111010$, $j = 58$ и $f_{58} = 1$, следовательно, троичный вектор $\boldsymbol{t}^{27} = -1101-$ представляет обязательную простую импликанту.

Утверждение 5. Предположим, что k-й элемент вектора $\boldsymbol{f}$ имеет трех соседей. Тогда вектор $\boldsymbol{t}^k$ представляет обязательную простую импликанту функции f, если и только если $\boldsymbol{n}^k \leqslant \boldsymbol{n}^j$ (вектор $\boldsymbol{n}^k$ поглощается вектором $\boldsymbol{n}^j$), где $\boldsymbol{b}^j = \boldsymbol{b}^k \oplus \boldsymbol{n}^k$.

Доказательство этого утверждения основано на том факте, что векторы $\boldsymbol{b}^k$ и $\boldsymbol{b}^j$ являются противолежащими элементами в интервале Int_k ранга 3 и вместе со своими соседями покрывают все восемь элементов этого интервала (рис. 1).

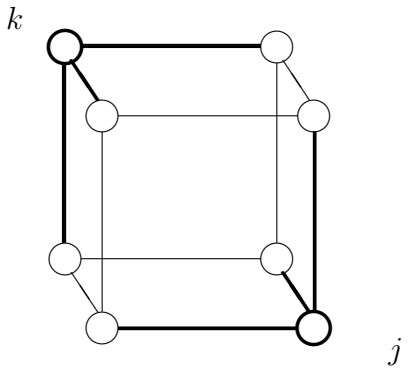


Рис. 1

Утверждение 6. Предположим, что k-й элемент вектора $\boldsymbol{f}$ имеет четырех соседей. Тогда вектор $\boldsymbol{t}^k$ представляет обязательную простую импликанту функции f, если и только если $\boldsymbol{n}^k \leqslant \boldsymbol{n}^j$ для трех (это минимальное число) значений индекса j, которые можно определить в соответствии с рис. 2.

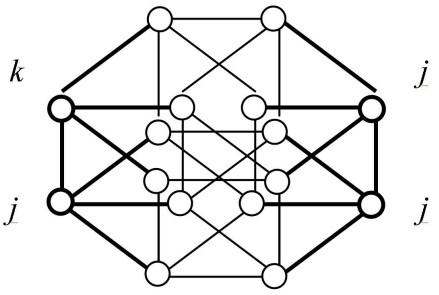


Рис. 2

Возникает вопрос о практической целесообразности проверки компонент вектора $\boldsymbol{f}$ на выполнение условий, сформулированных в утверждениях 4 – 6. Предположим, что задана булева функция от n переменных с вероятностью $1/2$ значения 1 в каждой компоненте, независимо друг от друга. Рассмотрим некоторую компоненту со значением 1 и k соседями. Насколько вероятно, что эта компонента определяет соответствующую простую импликанту ранга $n - k$?

Утверждение 7. Вероятность такого события равна $1/2^t$, где $t = 2^k - (k + 1)$.

Эта вероятность быстро стремится к нулю. Например, при $k = 2, 3, 4$ и 5 она равна $1/2$, $1/16$, $1/2048$ и $1/67\,208\,864$ соответственно, будучи независима от n.

Поэтому в предлагаемом эвристическом алгоритме анализу подвергаются лишь такие единичные компоненты вектора $\boldsymbol{f}$, число соседей у которых не превышает трех.

5. Алгоритм получения частичного решения

В предлагаемом алгоритме последовательно находятся импликанты высокого ранга. Результат фиксируется введением единиц в вектор решения $\boldsymbol{g}$ (сначала $\boldsymbol{g} = \boldsymbol{0}$), определением соответствующих им столбцов матрицы решения $\boldsymbol{D}$ и корректировкой вектора $\boldsymbol{f}^*$, представляющего множество еще не покрытых элементов характеристического множества M^1.

1. $\boldsymbol{g} := \boldsymbol{m}_0$, $\boldsymbol{f}^* := \boldsymbol{f} \setminus \boldsymbol{m}_0$. Так находятся импликанты ранга n (в данном примере это две импликанты, задаваемые векторами 000000 и 100110 и определяемые элементами вектора $\boldsymbol{f}$ с номерами 0 и 38).

2. Последовательно рассматриваются элементы вектора $\boldsymbol{f}$, отмеченные одновременно в векторах $\boldsymbol{m}_1$ и $\boldsymbol{f}^*$. Для очередного элемента f^k берется соответствующий ему набор $\boldsymbol{b}^k$, компонента которого, отмеченная единицей в столбце $\boldsymbol{n}^k$ матрицы соседства $\boldsymbol{N}$, заменяется на символ «—». Результат представляет простую импликанту ранга $n-1$. Соответственно корректируются векторы $\boldsymbol{g}$ и $\boldsymbol{f}^*$: в первый добавляется одна единица, а из второго удаляются две.

Так, в данном примере последовательно рассматриваются элементы с номерами $k = 18, 24, 48$ и 55, которым соответствуют наборы 010010, 011000, 110000 и 110111 и которые определяют простые импликанты ранга $n-1$: $01{-}010, 0110{-}0, 110{-}00, {-}10111$. Вектор решения принимает значение

$$g = 10000000\ 00000000\ 00100000\ 10000000\ 00000010\ 00000000\ 10000001\ 00000000$$

и соответственно (в результате удаления поглощаемых элементов они отмечены подчеркиванием) меняется значение вектора $\boldsymbol{f}^*$:

$$f^* = \underline{0}0010101\ 00100110\ 00\underline{0}01\underline{10}\ \underline{00}\underline{0}10010\ 0001\underline{00}\underline{0}\ 01010100\ \underline{0}\underline{000}\underline{0}\underline{00}\underline{0}\ 00111010.$$

3. На этом этапе рассматриваются последовательно элементы f^k с двумя соседями, отмеченные одновременно в векторах $\boldsymbol{m}_2$ и $\boldsymbol{f}^*$, и строятся импликанты ранга $n-2$.

Сначала находятся элементы f^k, удовлетворяющие условию утверждения 4. Соответствующие компоненты вектора $\boldsymbol{g}$ принимают значение 1, столбцы $\boldsymbol{d}^k$ матрицы $\boldsymbol{D}$ остаются равными столбцам $\boldsymbol{n}^k$ матрицы $\boldsymbol{N}$, и из вектора $\boldsymbol{f}^*$ удаляются единицы, покрываемые интервалами Int_k.

Если же условие утверждения 4 не выполняется, из двух соседей элемента f^k выбирается один сосед, желательно еще не покрытый, в столбце $\boldsymbol{d}^k$ остается лишь одна соответствующая единица и выполняется операция, предусмотренная для элемента с одним соседом.

В данном примере (он оказывается довольно простым) это приводит к окончательному решению — покрытию всех элементов характеристического множества M^1, получению пары векторов

$$g = 10010100\ 00000110\ 00101000\ 10010000\ 00000010\ 01000000\ 10000001\ 00001000,$$
$$f^* = 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000$$

и построению матрицы решения $\boldsymbol{D}$, получаемой из $\boldsymbol{N}$ удалением одной единицы в столбцах, отмеченных в векторе

$$\boldsymbol{m}_2 = 00010000\ 00100110\ 00001000\ 00010000\ 00010000\ 01000100\ 00000000\ 00001000,$$

и удалением всех единиц в столбцах, не отмеченных в векторе g :

$$D = \begin{matrix} 00010000\ 00000100\ 00000000\ 00010000\ 00000000\ 00000000\ 00000001\ 00000000 \\ 00000100\ 00000010\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000 \\ 00000000\ 00000000\ 00100000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000 \\ 00000000\ 00000010\ 00000000\ 00000000\ 00000000\ 00000000\ 10000000\ 00000000 \\ 00000100\ 00000000\ 00000000\ 10000000\ 00000000\ 01000000\ 00000000\ 00001000 \\ 00000000\ 00000000\ 00001000\ 00010000\ 00000000\ 00000000\ 00000000\ 00000000. \end{matrix}$$

4. Если вектор $\boldsymbol{f}^*$ остается отличным от нуля, рассматриваются элементы с тремя соседями, отмеченные одновременно в векторах $\boldsymbol{m}_3$ и $\boldsymbol{f}^*$. Находятся элементы, удовлетворяющие условию утверждения 5, в решение вводятся соответствующие импликанты ранга $n-3$. При невыполнении данного условия находятся определяемые этими элементами импликанты более высокого ранга: $n-1$ и $n-2$.

В результате описанной процедуры обработки элементов вектора $\boldsymbol{f}$, имеющих не более трех соседей, мы получаем множество импликант, образующих *частичное решение* S и вектор $\boldsymbol{f}^*$, представляющий *остаток* — совокупность не покрытых этим решением элементов множества M^1.

6. Итеративный алгоритм

Идея этого эвристического алгоритма состоит в следующем. Сначала он находит частичное решение для вектора $\boldsymbol{f}$, затем выполняет такую же операцию для остатка $\boldsymbol{f}^*$, дополняя множество получаемых импликант и соответственно упрощая вектор $\boldsymbol{f}^*$ (удаляя из него некоторые единицы). Если после упрощения $\boldsymbol{f}^*$ в нем остаются единицы, выполняется следующая итерация и так далее, пока $\boldsymbol{f}^*$ не станет равным нулю.

Получаемые при этом импликанты могут быть далеко не обязательными и даже не простыми. Поэтому в заключение они приводятся к простым (понижением ранга), а также устраняются получаемые дубли.

Продемонстрируем алгоритм на конкретном примере. Пусть $n = 19$ и каждый элемент вектора $\boldsymbol{f}$ принимает значение 1 с вероятностью $p = 1/4$. При этом предположении был сгенерирован случайный вектор $\boldsymbol{f}$ с 147 232 единицами и построена матрица соседства $\boldsymbol{N}$ с $2^{19} = 534\,288$ столбцами и следующим распределением столбцов по числу единиц в них ($N(i)$ столбцов содержат по i единиц):

$$\begin{array}{rcccccccccccccccccc} i & = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16, \\ N(i) & = & 296 & 2087 & 7180 & 16352 & 25357 & 29868 & 26913 & 19406 & 11379 & 5401 & 2087 & 690 & 172 & 35 & 8 & 1 & 0. \end{array}$$

При первой итерации алгоритма обрабатываются элементы множества M^1 с числом соседей $i = 0, 1, 2$ и 3, всего $296 + 2087 + 7180 + 16352 = 25915$ элементов. Находятся определяемые ими обязательные простые импликанты, общим числом $296 + 2082 + {} + 1974 + 80 = 4432$. Общее число найденных при первой итерации простых импликант (вместе с необязательными) равно 24 379. Они отмечаются в векторе решения $\boldsymbol{g}$, и покрываемые ими элементы удаляются из переменного вектора остатка $\boldsymbol{f}^*$ (вначале равного $\boldsymbol{f}$). Число единиц в векторе $\boldsymbol{f}^*$ становится равным 81 910.

При второй итерации рассматривается новая матрица $\boldsymbol{N}$, представляющая отношение соседства на элементах множества M^1, отмеченных в векторе остатка $\boldsymbol{f}^*$. В этом

векторе выявляются элементы, имеющие не более трех соседей в этом же векторе, находятся соответствующие ими импликанты. Векторы $\boldsymbol{g}$ и $\boldsymbol{f}^*$ получают новые значения. Если после этого $\boldsymbol{f}^* \neq \boldsymbol{0}$, выполняется следующая итерация.

Так для рассматриваемого примера выполняются четыре итерации, прежде чем вектор $\boldsymbol{f}^*$ становится равным $\boldsymbol{0}$ — все элементы множества M^1 оказываются покрытыми 61 477 импликантами со следующим распределением их по рангам (ранг – число импликант):

$$19 - 2\,458, 18 - 33\,668, 17 - 25\,237, 16 - 114.$$

Как уже говорилось, не все эти импликанты оказываются простыми. Поэтому в заключение выполняются две процедуры приведения полученного решения к корректному виду. Первая из них упрощает импликанты, обращаясь к исходной матрице соседства $\boldsymbol{N}$ и удаляя из импликант некоторые литералы, если это возможно. Она приводит к новому распределению импликант по рангам:

$$19 - 296, 18 - 20\,933, 17 - 38\,617, 16 - 1\,631.$$

Вторая процедура устраняет дубли среди полученных импликант, в результате чего число последних сокращается до 60 972, а распределение их по рангам принимает следующий вид:

$$19 - 296, 18 - 20\,933, 17 - 38\,353, 16 - 1\,390.$$

7. Компьютерные эксперименты

Изложенный выше итеративный алгоритм был программно реализован на C++ и испытан на компьютере (Pentium IV, 2.8 GHz). Чтобы было удобней работать с булевыми векторами, в которых перечислены соседи для рассматриваемых элементов вектора $\boldsymbol{f}$, матрица соседства $\boldsymbol{N}$ предварительно транспонировалась.

Эксперименты проводились на множестве псевдослучайных булевых функций, с двумя параметрами: *числом переменных* n и *плотностью единиц* r, определяющей ожидаемое число единиц q в представляющем булеву функцию f векторе $\boldsymbol{f}$: $r = 32q/2^n$. Например, при $r = 16$ рассматривается абсолютно случайная булева функция, принимающая на каждом элементе булева пространства значение 1 с вероятностью $1/2$.

Прежде всего, были определены границы практической применимости предложенного алгоритма в пространстве параметров n и r. Дело в том, что при достаточно большом значении параметра r алгоритм может прекратить свою работу после некоторого числа итераций, поскольку числа $N(0), N(1), N(2)$ и $N(3)$ могут оказаться равными нулю, в то время как вектор $\boldsymbol{f}^*$ будет оставаться отличным от $\boldsymbol{0}$.

Например, если $n = 17$ и $r = 15$, алгоритм останавливается после восьми итераций, найдя всего 636 импликант. Но при $n = 17$ и $r = 14$ он находит после 90 итераций 20 077 импликант, полностью покрывающих множество M^1 (после последующего упрощения этих импликант и устранения дублей их число сокращается до 19 811). Следовательно, пара $(n, r) = (17, 14)$ является элементом верхней границы области применения алгоритма.

В табл. 1 представлены основные характеристики этой границы, полученные экспериментально. Строки таблицы соответствуют числу переменных n (от 4 до 23) и соответствующим им максимальным значениям параметра r, при которых программа работает корректно. Кроме того, в строках показаны следующие величины:

N — число единиц в векторе $\boldsymbol{f}$ (число импликант в совершенной ДНФ функции f);

C — число импликант в полученной ДНФ;
S — длина полученной ДНФ (сумма рангов импликант);
Q_k — число импликант ранга k в полученной ДНФ (при $k = n, n-1, n-2, n-3, n-4$);
It — число итераций;
T – затраченное время, в с.

Таблица 1

Экспериментальные характеристики границы области применения алгоритма

n	r	N	C	S	Q_n	Q_{n-1}	Q_{n-2}	Q_{n-3}	Q_{n-4}	It	T
4	16	5	3	10	1	2	0	0	0	1	0,00
5	16	16	8	27	0	3	5	0	0	1	0,00
6	16	31	14	62	1	4	9	0	0	1	0,00
7	16	62	31	169	1	12	18	0	0	1	0,00
8	16	138	58	360	2	18	28	10	0	2	0,00
9	16	274	107	744	0	15	72	20	0	3	0,00
10	16	556	194	1513	0	14	127	53	0	3	0,00
11	16	1083	379	3355	1	35	250	93	0	4	0,01
12	16	2203	735	7127	0	31	456	242	6	5	0,03
13	16	4337	1414	15057	0	45	835	526	8	7	0,09
14	16	8734	2780	32266	0	64	1579	1116	21	12	0,35
15	15	16404	5313	67324	1	168	3258	1858	27∗	13	1,01
16	14	31021	10181	139827	1	412	6671	3073	24	13	3,12
17	14	61150	19811	291507	1	684	12842	6224	60	90	24,37
18	13	114347	38007	500357	4	1809	26670	9476	48	21	42,26
19	12	212620	72343	1220742	7	4787	53693	13822	34	12	148
20	11	392995	137215	2462996	34	12639	104999	19505	38	10	610
21	11	786345	270642	5121875	54	21491	207248	41776	73	18	2499
22	10	1442274	512529	10255122	127	58852	399029	54478	43	10	8858
23	9	2620069	966357	20386490	481	158666	740594	66597	19	8	31064

Табл. 2 отображает поведение алгоритма при числе переменных $n = 17$. Видно, как растет число итераций It и время решения T при возрастании плотности единиц r. При этом распределение полученных импликант по рангам быстро изменяется в пользу импликант меньшего ранга.

Таблица 2

Поведение алгоритма при числе переменных $n = 17$

n	r	N	C	S	Q_{17}	Q_{16}	Q_{15}	Q_{14}	Q_{13}	It	T
17	2	12285	7856	127689	2283	5283	290	0	0	2	0,27
17	4	20427	11117	177205	1147	8162	1802	6	0	2	0,53
17	6	28594	13761	215604	417	8406	4887	51	0	3	1,90
17	8	36507	15621	240722	135	6370	8883	233	0	3	4,04
17	10	44756	17348	263178	41	3864	12456	986	1	4	6,49
17	12	52999	18691	279341	6	1880	13899	2896	10	7	8,43
17	14	61150	19811	291507	1	684	12842	6224	60	90	24,52

Поведение алгоритма при максимально возможном для него числе переменных ($n = 24$) показано в табл. 3. При увеличении плотности единиц r на единицу время решения T растет более чем вдвое, достигая 6,8 ч при $r = 4$.

Таблица 3

Поведение алгоритма при числе переменных $n = 24$

n	r	N	C	S	Q_{24}	Q_{23}	Q_{22}	Q_{21}	It	T
24	1	1047350	685881	15982597	223163	446889	15829	0	2	1693,45
24	2	1571532	919682	21231157	148570	701045	70035	32	2	4068,76
24	3	2095590	1124293	25759214	85794	853387	184905	207	3	10695,15
24	4	2619724	1297946	29532155	44585	889335	362864	1162	3	24348,00

Заключение

В предлагаемом алгоритме минимизации булевых функций многих переменных (до 24) реализованы следующие идеи. 1) Роль элементарных операндов играют булевы векторы с 2^n компонентами, представляющие произвольные булевы функции n переменных. 2) Строится булева матрица соседства $\boldsymbol{N}$ размером $n \times 2^n$, отображающая структуру характеристического множества M^1. 3) С ее помощью быстро находятся простые импликанты четырех старших рангов, покрывающие часть множества M^1. 4) Структура остатка представляется новой матрицей $\boldsymbol{N}$, находятся новые импликанты высокого ранга и т. д. Итерации прекращаются, когда множество M^1 становится пустым. 5) В заключение полученные импликанты доводятся до простых и устраняются дубли. Алгоритм реализуется эффективной программой, которая может оказаться полезной при решении систем булевых уравнений, верификации логических схем и решении других трудоемких задач теории булевых функций.

ЛИТЕРАТУРА

1. *Закревский А. Д.* Логический синтез каскадных схем. М., 1981.
2. *Zakrevskij A. D.* Parallel operations over neighbors in Boolean space //Proceedings of the Sixth International Conference CAD DD-07. Minsk, 2007. V. 2. P. 613.
3. *Закревский А. Д.* Программирование вычислений в многомерном булевом пространстве // 7-я Российская конф. с международным участием «Новые информационные технологии в исследовании сложных структур». Томск, 2008.

2009 Теоретические основы прикладной дискретной математики №1(3)
DOI 10.17223/20710410/3/2
УДК 519.7

БЕНТ-ФУНКЦИИ: РЕЗУЛЬТАТЫ И ПРИЛОЖЕНИЯ. ОБЗОР РАБОТ[1]

Н. Н. Токарева

Институт математики им. С. Л. Соболева СО РАН, г. Новосибирск

E-mail: tokareva@math.nsc.ru

Приводится краткий обзор основных результатов в области бент-функций. Рассматриваются их теоретические и практические приложения.

Ключевые слова: *булева функция, АНФ, преобразование Уолша—Адамара, нелинейность, бент-функция.*

Введение

Данный обзор посвящен результатам в области бент-функций и их приложениям. Логическим продолжением этой работы следует считать обзор [16], в котором будут рассмотрены различные обобщения бент-функций и история их возникновения.

Мера нелинейности является важной характеристикой булевой функции. Линейность часто свидетельствует о простой (в определенном смысле) структуре этой функции и, как правило, представляет собой богатый источник информации о многих других ее свойствах. Задача построения булевых функций, обладающих нелинейными свойствами, естественным образом возникает во многих областях дискретной математики. И часто (что является типичной ситуацией в математике) наибольший интерес вызывают те функции, для которых эти свойства экстремальны. Такие булевы функции называются *максимально нелинейными* или *бент-функциями*.

Первой работой, посвященной бент-функциям, принято считать статью О. Ротхауса 1976 г. [110], хотя эти специальные булевы функции были введены им еще в 60-х годах XX века [109]. В настоящее время можно говорить уже о *теории бент-функций*.

Приведем ряд определений и обозначений. Пусть

$\oplus$ — сложение по модулю 2 (или XOR);

n — натуральное число;

$\boldsymbol{v} = (v_1, \ldots, v_n)$ — двоичный вектор;

$\mathbb{Z}_2^n$ — множество всех двоичных векторов длины n;

$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = u_1 v_1 \oplus \ldots \oplus u_n v_n$ — скалярное произведение векторов;

$f : \mathbb{Z}_2^n \to \mathbb{Z}_2$ — булева функция от n переменных;

$\boldsymbol{f}$ — вектор значений длины 2^n функции f. Будем считать, что аргументы функции (т. е. векторы длины n) перебираются в лексикографическом порядке;

$\mathrm{dist}(f, g)$ — *расстояние Хэмминга* между функциями f и g, т. е. число позиций, в которых различаются векторы $\boldsymbol{f}$ и $\boldsymbol{g}$.

Известно, что каждая булева функция однозначно может быть задана своей *алгебраической нормальной формой* (АНФ), а именно представлена в виде

$$f(\boldsymbol{v}) = \left(\bigoplus_{k=1}^{n} \bigoplus_{i_1,\ldots,i_k} a_{i_1,\ldots,i_k} \, v_{i_1} \cdot \ldots \cdot v_{i_k} \right) \oplus a_0,$$

[1]Исследование выполнено при финансовой поддержке интеграционного проекта СО РАН № 35 «Древовидный каталог математических интернет-ресурсов mathtree.ru», РФФИ (проекты 07-01-00248, 08-01-00671, 09-01-00528-а) и Фонда содействия отечественной науке.

где при каждом k индексы $i_1, \ldots, i_k$ различны и в совокупности пробегают все k-элементные подмножества множества $\{1, \ldots, n\}$. Коэффициенты $a_{i_1,\ldots,i_k}$, a_0 принимают значения 0 или 1. В русскоязычной литературе АНФ также называют *полиномом Жегалкина*. Напомним, что *степень нелинейности* $\deg(f)$ булевой функции f — это число переменных в самом длинном слагаемом ее АНФ. Функция называется *аффинной*, *квадратичной*, *кубической* и т. д., если ее степень равна соответственно 1, 2, 3 и т. д. Каждая аффинная функция от n переменных $v_1, \ldots, v_n$ имеет вид $\langle \boldsymbol{u}, \boldsymbol{v} \rangle \oplus a$ для подходящих вектора $\boldsymbol{u}$ и константы a.

Максимально нелинейной называется булева функция от n переменных (n любое) такая, что расстояние Хэмминга N_f от данной функции до множества всех аффинных функций является максимально возможным. Величину N_f называют *нелинейностью* булевой функции. В случае четного n максимально возможное значение нелинейности равно $2^{n-1} - 2^{(n/2)-1}$. В случае нечетного n точное значение максимального расстояния неизвестно (поиск этого значения или его оценок представляет весьма любопытную и сложную комбинаторную задачу [89, 78]). Термин «максимально нелинейная функция» принят в русскоязычной литературе, тогда как в англоязычной широкое распространение получил термин «бент-функция» (от англ. bent — изогнутый, наклоненный). Аналогия между терминами не полная. При четном числе переменных n бент-функции и максимально нелинейные функции совпадают, а при нечетном n бент-функции (в отличие от максимально нелинейных) не существуют.

Преобразованием Уолша—Адамара булевой функции f от n переменных называется целочисленная функция W_f, заданная на множестве $\mathbb{Z}_2^n$ равенством

$$W_f(\boldsymbol{v}) = \sum_{\boldsymbol{u} \in \mathbb{Z}_2^n} (-1)^{\langle \boldsymbol{u}, \boldsymbol{v} \rangle \oplus f(\boldsymbol{u})}.$$

Справедливо равенство Парсеваля, $\sum\limits_{\boldsymbol{v} \in \mathbb{Z}_2^n} \left(W_f(\boldsymbol{v})\right)^2 = 2^{2n}$. Поскольку число всех коэффициентов $W_f(\boldsymbol{v})$ равно 2^n, из равенства следует, что максимум модуля коэффициента Уолша—Адамара не может быть меньше величины $2^{n/2}$.

Нелинейность N_f произвольной функции f тесно связана с ее коэффициентами Уолша—Адамара, а именно $N_f = 2^{n-1} - \frac{1}{2} \max\limits_{\boldsymbol{v} \in \mathbb{Z}_2^n} |W_f(\boldsymbol{v})|$. Очевидно, что чем меньше максимум модуля коэффициента $W_f(\boldsymbol{v})$, тем выше нелинейность функции f.

Бент-функцией называется булева функция от n переменных (n четно) такая, что модуль каждого коэффициента Уолша—Адамара этой функции равен $2^{n/2}$. Другими словами, f — бент-функция, если максимум модуля $W_f(\boldsymbol{v})$ достигает своего минимального возможного значения. В силу равенства Парсеваля это имеет место, только если модули всех коэффициентов Уолша—Адамара этой функции совпадают и равны $2^{n/2}$. Таким образом, эквивалентность определению максимально нелинейной функции (при четном n) становится очевидной.

1. Приложения

Впечатляют масштабы исследования бент-функций. В настоящее время несколько сотен математиков и инженеров по всему миру регулярно публикуют свои статьи по этой тематике. Результаты обсуждаются на таких международных конференциях, как EUROCRYPT, CRYPTO, ASIACRYPT, INDOCRYPT, SETA, FSE, AAECC, ISIT, ITW, BFCA, ACCT, SIBECRYPT, МаБИТ и многих других. Счет общего числа публикаций о бент-функциях (и близких вопросах) идет на тысячи. К сожалению, публикаций на русском языке известно не так уж много — всего несколько десятков.

Актуальность исследования бент-функций подтверждается их многочисленными теоретическими и практическими приложениями в комбинаторике, алгебре, теории кодирования, теории информации, теории символьных последовательностей, криптографии и криптоанализе. Приведем (далеко не полную) серию таких примеров.

Классическая комбинаторная задача построения *матриц Адамара* порядка N, известная с 1893 г., в случае $N = 2^n$ (n четно) при некоторых ограничениях сводится к задаче построения бент-функций от n переменных [110] (см. далее теорему 1).

В теории конечных групп построение *элементарных адамаровых разностных множеств* специального вида эквивалентно построению максимально нелинейных булевых функций (см. [8] и теорему 5).

В теории кодирования широко известна задача определения радиуса покрытия произвольного *кода Рида—Маллера*, которая эквивалентна (в случае кодов первого порядка) поиску наиболее нелинейных булевых функций [78, 89]. В теории оптимальных кодов специальные семейства квадратичных бент-функций определяют класс *кодов Кердока* [79], обладающих исключительным свойством: вместе с растущим кодовым расстоянием (при увеличении длины кода) каждый код Кердока имеет максимально возможную мощность (см. [13, 59]). Этим свойством коды Кердока «обязаны» экстремальной нелинейности бент-функций. Отметим, что задача построения таких семейств бент-функций, задающих код Кердока, несложно переводится в задачу поиска *ортогональных расщеплений* (orthogonal spreads) в конечном векторном пространстве [77], что представляется элегантным примером связи бент-функций с экстремальными геометрическими объектами. Другим примером из теории кодирования служат так называемые *бент-коды* — линейные двоичные коды, каждый из которых определенным образом строится из некоторой бент-функции [40]. В принципе, тем же способом можно строить линейные коды из любых булевых функций, но только бент-коды имеют максимально возможные кодовые размерности.

Семейства *бент-последовательностей* из элементов $+1$ и -1, построенные на основе бент-функций, имеют предельно низкие значения как взаимной корреляции, так и автокорреляции (достигают нижней границы Велча) [99]. Поэтому такие семейства успешно применяются в коммуникационных системах коллективного доступа [102]. Генераторы бент-последовательностей легко инициализируются случайным образом и могут быстро перестраиваться с одной последовательности на другую. Этот факт используется в работе со стандартом CDMA — Code Division Multiple Access (множественный доступ с кодовым разделением каналов) — одним из двух стандартов для цифровых сетей сотовой связи в США. Отметим здесь же, что в системах CDMA для предельного снижения отношения пиковой и средней мощностей сигнала (peak-to-average power ratio) используются так называемые *коды постоянной амплитуды* (constant-amplitude codes). И например, четверичные такие коды можно построить с помощью обобщенных булевых бент-функций [111]. Не обходится без бент-функций или их аналогов и в квантовой теории информации (см., например, [106]).

Бент-функцию можно определить как функцию, которая крайне плохо аппроксимируется аффинными функциями. Это базовое свойство бент-функций используется в криптографии. В блочных и поточных шифрах бент-функции и их векторные аналоги способствуют предельному повышению стойкости этих шифров к линейному [90] и дифференциальному [28] методам криптоанализа. Стойкость достигается за счет использования сильно нелинейных булевых функций в S-блоках (важнейших компонентах современных шифров) [21, 96] (см., например, шифр CAST [49]). Бент-функции

и их обобщения находят свое применение также в схемах аутентификации [45], хэш-функциях и псевдослучайных генераторах [10].

Широко исследуются различные обобщения, подклассы и надклассы бент-функций, такие, как *платовидные функции* [8], *частично бент-функции* [8], *частично определенные бент-функции* [8], *q-значные бент-функции* [2, 83], *обобщенные булевы бент-функции* [111], *полу-бент-функции* [53], *бент-функции на конечной абелевой группе* [6, 14, 44], *однородные бент-функции* [105], *гипер-бент-функции* [114], *$\mathbb{Z}$-бент-функции* [68], *нега-бент-функции* [101], *k-бент-функции* [15] и др. С одной стороны, эти исследования мотивированы высокой сложностью задачи описания бент-функций и являются попытками перехода к более общим (или более частным) ее постановкам — в надежде на частичное решение основной проблемы. С другой стороны, интерес к обобщениям постоянно стимулируется новыми запросами со стороны приложений.

Обзоры некоторых результатов о бент-функциях можно найти в замечательной российской монографии 2004 г. О. А. Логачева, А. А. Сальникова и В. В. Ященко [8], статье немецких криптографов Х. Доббертина и Г. Леандера [67] 2004 г., главах [40, 41] французского математика и криптографа К. Карле, написанных для готовящейся к печати книги «Boolean Methods and Models» (2008 г.). См. также более ранние работы Ю. В. Кузнецова и С. А. Шкарина [4] 1996 г., Дж. Ф. Диллона [62] 1972 г. и др. Обобщениям бент-функций будет посвящен отдельный обзор автора [16].

Однако любой обзор в этой области очень быстро устаревает и a priori неполон.

2. Результаты

Бент-функции, как уже упоминалось выше, были введены О. Ротхаусом еще в 60-х годах XX века. В работе [110] были установлены базовые свойства таких функций и предложены их простейшие конструкции. Дж. Диллон [62] и Р. Л. МакФарланд [91] рассматривали бент-функции в связи с разностными множествами. В настоящее время известны серии конструкций бент-функций, но тем не менее класс всех бент-функций от n переменных (обозначим его через $\mathfrak{B}_n$) до сих пор не описан, для мощности этого класса не найдена асимптотика и не установлено даже приемлемых нижних и верхних оценок. Известным результатам и открытым вопросам в области бент-функций посвящен этот раздел.

2.1. Критерии и свойства

Всюду далее n предполагается четным числом.

Напомним, что *матрицей Адамара* называется квадратная $k \times k$-матрица A с элементами ± 1, такая, что $AA^T = kE$, где E — единичная матрица. Строки и столбцы матрицы размера $2^n \times 2^n$ занумеруем векторами $\boldsymbol{u}, \boldsymbol{v}$ длины n. Справедлива [110]

Теорема 1. Следующие утверждения эквивалентны:

(i) булева функция f от n переменных является бент-функцией;

(ii) матрица $A = (a_{\boldsymbol{u},\boldsymbol{v}})$, где $a_{\boldsymbol{u},\boldsymbol{v}} = \frac{1}{2^{n/2}} W_f(\boldsymbol{u} \oplus \boldsymbol{v})$, является матрицей Адамара;

(iii) матрица $D = (d_{\boldsymbol{u},\boldsymbol{v}})$, где $d_{\boldsymbol{u},\boldsymbol{v}} = (-1)^{f(\boldsymbol{u} \oplus \boldsymbol{v})}$, является матрицей Адамара;

(iv) для любого ненулевого вектора $\boldsymbol{u}$ функция $f(\boldsymbol{v}) \oplus f(\boldsymbol{v} \oplus \boldsymbol{u})$ *сбалансирована*, т. е. принимает значения 0 и 1 одинаково часто.

Пункт (iv) теоремы носит название *критерия распространения* $\mathrm{PC}(n)$ *порядка* n. В качестве важного свойства бент-функций можно отметить следующий факт [110], согласно [5] полученный В. А. Елисеевым и О. П. Степченковым еще в 1962 г.

Теорема 2. Степень нелинейности $\deg(f)$ любой бент-функции f от $n \geqslant 4$ переменных не превосходит числа $n/2$.

Аффинная функция, как нетрудно видеть, не может быть бент-функцией. Сразу отметим, что бент-функции любой другой возможной степени существуют. Например, квадратичной бент-функцией при любом четном n является функция

$$f(v_1, \ldots, v_n) = v_1 v_2 \oplus v_3 v_4 \oplus \ldots \oplus v_{n-1} v_n. \tag{1}$$

Интересно, что любую другую квадратичную бент-функцию g можно получить из f аффинным преобразованием. Приведем необходимое определение. Булевы функции f и g от n переменных *аффинно эквивалентны*, если существуют невырожденная $n \times n$ матрица A, векторы $\boldsymbol{b}, \boldsymbol{c}$ длины n и константа $\lambda \in \mathbb{Z}_2$, такие, что

$$g(\boldsymbol{v}) = f(A\boldsymbol{v} \oplus \boldsymbol{b}) \oplus \langle \boldsymbol{c}, \boldsymbol{v} \rangle \oplus \lambda.$$

Согласно [52] (см. [9, 62]), выполняется

Теорема 3. Любая квадратичная бент-функция от n переменных аффинно эквивалентна функции (1).

Для бент-функций степени 3 и выше подобных результатов нет. Справедлива

Теорема 4. Класс $\mathfrak{B}_n$ бент-функций замкнут относительно
(i) любого невырожденного аффинного преобразования переменных;
(ii) прибавления любой аффинной функции.

В силу теоремы 4 имеет смысл вопрос об аффинной классификации бент-функций, который для функций степени $\geqslant 3$ пока остается открытым. Подробнее о методах аффинной и линейной классификации булевых функций можно прочитать в [18].

Часто с бент-функцией f связывают так называемую *дуальную булеву функцию* $\widetilde{f}$ от n переменных, которая определяется равенством $W_f(\boldsymbol{v}) = 2^{n/2}(-1)^{\widetilde{f}(\boldsymbol{v})}$. Определение корректно, поскольку $W_f(\boldsymbol{v}) = \pm 2^{n/2}$ для каждого вектора $\boldsymbol{v}$. Несложно доказать, что булева функция $\widetilde{f}$ является бент-функцией. Справедливо $\widetilde{\widetilde{f}} = f$. Отметим, что если $\deg(f) = n/2$, то степень $\widetilde{f}$ также максимальна: $\deg(\widetilde{f}) = n/2$. Самодуальные бент-функции, т. е. такие, что $f = \widetilde{f}$, изучались в [43].

Дуальные бент-функции потребуются далее в теореме 14.

2.2. Характеризации бент-функций

Рассмотрим ряд попыток найти бент-функциям комбинаторные или алгебраические «эквиваленты».

С самого начала бент-функции изучались в связи с разностными множествами [62]. Пусть конечная абелева группа G имеет порядок v и дана в аддитивной записи. Подмножество $D \subseteq G$ мощности k называется *разностным множеством* с параметрами (v, k, λ), если каждый ненулевой элемент $g \in G$ можно представить в виде $g = b - d$ ровно λ способами, где b, d — элементы множества D. Справедлива [62]

Теорема 5. Булева функция f от n переменных является бент-функцией, если и только если множество $D = \{\, (\boldsymbol{v}, f(\boldsymbol{v})) \mid \boldsymbol{v} \in \mathbb{Z}_2^n \,\}$ является разностным множеством с параметрами $(2^{n+1}, 2^n, 2^{n-1})$ в аддитивной группе $\mathbb{Z}_2^{n+1}$.

Разностные множества с приведенными в теореме 5 параметрами называются *элементарными адамаровыми*. Примеры таких множеств были известны еще до появления бент-функций [62].

Известно [17], что разностные множества тесно связаны с блок-схемами. Напомним, что *блок-схемой* с параметрами (v, k, λ) называется система k-элементных подмножеств (или *блоков*) v-элементного множества, такая, что каждая пара различных

элементов содержится ровно в λ блоках. Блок-схема *симметрична*, если число блоков равно числу элементов, т. е. равно v. Теорема 5 имеет следующий эквивалентный вид.

Теорема 6. Булева функция f от n переменных является бент-функцией, если и только если система множеств $D_{\boldsymbol{z}} = D \oplus \boldsymbol{z}$, где вектор $\boldsymbol{z}$ пробегает $\mathbb{Z}_2^{n+1}$, является симметричной блок-схемой с параметрами $(2^{n+1}, 2^n, 2^{n-1})$.

В. В. Ященко [19] в 1997 г. предложил следующее описание класса бент-функций. В его основе лежит тот факт, что любая булева функция f от n переменных может быть представлена в виде *линейного разветвления*

$$f(\boldsymbol{u'}, \boldsymbol{u''}) = \langle \boldsymbol{u'}, h(\boldsymbol{u''}) \rangle \oplus g(\boldsymbol{u''}), \text{ где } \boldsymbol{u'} \in \mathbb{Z}_2^r, \boldsymbol{u''} \in \mathbb{Z}_2^k \tag{2}$$

для подходящих чисел r и k таких, что $n = r + k$, отображения $h : \mathbb{Z}_2^k \to \mathbb{Z}_2^r$ и булевой функции g от k переменных. Максимально возможное значение r в таком представлении называется *индексом линейности* булевой функции f.

Подмножество M пространства $\mathbb{Z}_2^n$ называется *бент-множеством*, если его мощность равна $2^{2\ell}$ при некотором ℓ и для любого ненулевого вектора $\boldsymbol{z} \in \mathbb{Z}_2^n$ множество $M \cap (\boldsymbol{z} \oplus M)$ либо пусто, либо имеет четную мощность.

Пара $(g; M)$, где g — булева функция от k переменных, M — бент-множество, называется *частичной бент-функцией*, если для любого $\boldsymbol{v'} \in \mathbb{Z}_2^r$ и ненулевого $\boldsymbol{v''} \in \mathbb{Z}_2^k$ функция $g(\boldsymbol{u''}) \oplus g(\boldsymbol{u''} \oplus \boldsymbol{v''})$ сбалансирована на множестве $(\boldsymbol{v'}, \boldsymbol{v''}) \oplus M$.

Теорема 7. Булева функция f вида (2) является бент-функцией тогда и только тогда, когда $n > 2r$ и для любого вектора $\boldsymbol{u'} \in \mathbb{Z}_2^r$ выполняются условия:
(i) мощность множества $h^{-1}(\boldsymbol{u'})$ равна 2^{n-2r};
(ii) множество $h^{-1}(\boldsymbol{u'})$ является бент-множеством;
(iii) пара $(g; h^{-1}(\boldsymbol{u'}))$ является частичной бент-функцией.

Позднее в 2004 г. К. Карле независимо предложил конструкцию бент-функций, представляющую собой частный случай данного описания (см. далее теорему 17).

Приведем геометрическое описание класса бент-функций, которое предложили в 1998 г. К. Карле и Ф. Гуилло [47] (см. также более раннюю работу [46]).

Пусть f — булева функция от n переменных. Пусть $\mathrm{Ind}_S : \mathbb{Z}_2^n \to \mathbb{Z}_2$ — характеристическая функция подмножества $S \subseteq \mathbb{Z}_2^n$, т. е. Ind_S принимает значение 1 на элементах из S и значение 0 на остальных элементах.

Теорема 8. Функция f является бент-функцией тогда и только тогда, когда существуют подпространства $E_1, \ldots, E_k$ размерности $n/2$ или $(n/2)+1$ пространства $\mathbb{Z}_2^n$ и ненулевые целые числа $m_1, \ldots, m_k$, такие, что для любого $\boldsymbol{v} \in \mathbb{Z}_2^n$ выполняется

$$\sum_{i=1}^{k} m_i \mathrm{Ind}_{E_i}(\boldsymbol{v}) = \pm 2^{(n/2)-1} \mathrm{Ind}_{\{\boldsymbol{0}\}}(\boldsymbol{v}) + f(\boldsymbol{v}).$$

Авторы [47] вводят ограничения на способ выбора пространств $E_1, \ldots, E_k$, при которых такой выбор становится единственным для каждой бент-функции. Таким образом, можно говорить об однозначности такого представления.

Другой подход предложили в 1999 г. А. Бернаскони и Б. Коденотти [26], затем к ним присоединился и Дж. Ван-дер-Кам [27].

Пусть f — булева функция от n переменных. Через $\mathrm{supp}(f)$ обозначим ее *носитель*, т. е. множество всех двоичных векторов длины n, на которых функция f принимает

значение 1. Рассмотрим *граф Кэли* $G_f = G(\mathbb{Z}_2^n, \mathrm{supp}(f))$ булевой функции f. Вершинами графа являются все векторы длины n. Две вершины $\boldsymbol{u}$, $\boldsymbol{v}$ соединяются ребром, если вектор $\boldsymbol{u} \oplus \boldsymbol{v}$ принадлежит множеству $\mathrm{supp}(f)$.

Граф G называется *сильно регулярным* (strongly regular), если существуют неотрицательные целые числа λ, μ такие, что для любых двух вершин x, y общее число смежных им вершин равно λ или μ в зависимости от того, соединены вершины x, y ребром или нет. В работе [27] доказана

Теорема 9. Булева функция f является бент-функцией тогда и только тогда, когда граф G_f является сильно регулярным, причем $\lambda = \mu$.

С. В. Агиевич в 2000 г. [22] установил биекцию между множеством всех бент-функций от n переменных и множеством *бент-прямоугольников*.

Пусть f — булева функция от n переменных, $n = r + k$. Вектор $\boldsymbol{W}_f$ коэффициентов Уолша—Адамара назовем *спектральным вектором* функции f. Представим двоичный вектор $\boldsymbol{f}$ в виде $\boldsymbol{f} = (\boldsymbol{f}_{(1)}, \ldots, \boldsymbol{f}_{(2^r)})$, где каждый вектор $\boldsymbol{f}_{(i)}$ имеет длину 2^k. Пусть $f_{(i)}$ — булева функция от k переменных, для которой $\boldsymbol{f}_{(i)}$ является вектором значений, $i = 1, \ldots, 2^r$. Свяжем с функцией f матрицу $\mathcal{M}_f$ размера $2^r \times 2^k$, строками которой являются спектральные векторы $\boldsymbol{W}_{f_{(1)}}, \ldots, \boldsymbol{W}_{f_{(2^r)}}$.

Матрица размера $2^r \times 2^k$ называется *бент-прямоугольником*, если каждая ее строка и каждый столбец, домноженный на $2^{r-(n/2)}$, являются спектральными векторами для подходящих булевых функций. Согласно [22], выполняется

Теорема 10. Булева функция f является бент-функцией тогда и только тогда, когда матрица $\mathcal{M}_f$ является бент-прямоугольником.

Данный подход позволил [22] дать описание всех бент-функций от шести переменных (см. далее) и получить алгоритм построения специального класса бент-функций от произвольного числа переменных n. В работе [24] С. В. Агиевич исследует соответствие между бент-прямоугольниками и регулярными q-значными бент-функциями [83], описывает аффинные трансформации первых и переводит на язык бент-прямоугольников основные конструкции бент-функций. Дальнейшее развитие этого подхода представляется весьма перспективным.

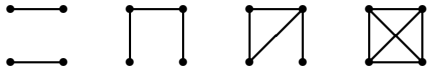
Здесь перечислены лишь некоторые возможные характеризации бент-функций.

2.3. Бент-функции от малого числа переменных

Задача описания всех бент-функций от n переменных решена лишь при малых значениях n. Приведем эти результаты.

$n = 2$. Функция $v_1 v_2$ является представителем единственного класса аффинной эквивалентности. Класс $\mathfrak{B}_2$ состоит из восьми функций. Это все функции, векторы значений которых содержат нечетное число единиц.

$n = 4$. Множество $\mathfrak{B}_4$ состоит из 896 булевых функций, причем каждая функция является квадратичной. Все бент-функции от четырех переменных аффинно эквивалентны функции $v_1 v_2 \oplus v_3 v_4$. Множество $\mathfrak{B}_4$ можно разделить на 28 классов по 32 функции. Алгебраические нормальные формы функций из каждого класса обладают одинаковой квадратичной частью, произвольной линейной частью и любым свободным членом. Если рассмотреть граф на множестве переменных, а ребрами соединить те вершины, которые образуют слагаемое в квадратичной части АНФ функции, то эти 28 типов можно задать следующим образом:



(3) (12) (12) (1)

Под каждым графом указано число типов, которые он определяет. Например, имеется 3 типа квадратичной части, состоящей из двух слагаемых: $v_1v_2 \oplus v_3v_4$, $v_1v_3 \oplus v_2v_4$, $v_1v_4 \oplus v_2v_3$, и только один тип из шести слагаемых.

$n = 6$. Аффинная классификация бент-функций от 6 переменных была получена еще в работе О. Ротхауса [110]: множество $\mathfrak{B}_6$ состоит из четырех классов аффинной эквивалентности, представителями которых являются следующие функции:

$v_1v_2 \oplus v_3v_4 \oplus v_5v_6$,

$v_1v_2v_3 \oplus v_1v_4 \oplus v_2v_5 \oplus v_3v_6$,

$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_1v_2 \oplus v_1v_4 \oplus v_2v_6 \oplus v_3v_5 \oplus v_4v_5$,

$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4v_6 \oplus v_1v_4 \oplus v_2v_6 \oplus v_3v_4 \oplus v_3v_5 \oplus v_3v_6 \oplus v_4v_5 \oplus v_4v_6$.

В работе [113] приводится подобная алгебраическая классификация. Пусть $GF(2^6) = \{0, 1, \alpha, \alpha^2, \ldots, \alpha^{62}\}$, где α — корень многочлена $x^6 + x + 1$. Пусть булева функция отождествляется с функцией $f(\boldsymbol{v}) : GF(2^6) \to GF(2)$, где $\boldsymbol{v}$ рассматривается как элемент поля $GF(2^6)$. Тогда в качестве представителей классов аффинной эквивалентности множества $\mathfrak{B}_6$ можно выбрать функции: $\mathrm{tr}(\boldsymbol{v}^3 + \alpha^5\boldsymbol{v}^5)$, $\mathrm{tr}(\alpha^3\boldsymbol{v}^7 + \boldsymbol{v}^9)$, $\mathrm{tr}(\alpha\boldsymbol{v}^3 + \alpha^6\boldsymbol{v}^7 + \alpha^{60}\boldsymbol{v}^{13})$, $\mathrm{tr}(\boldsymbol{v}^7 + \alpha\boldsymbol{v}^9 + \boldsymbol{v}^{21})$, где tr — функция следа из $GF(2^6)$ в $GF(2)$.

Дж. Диллоном [62] (см. также [30]) было показано, что любая бент-функция от шести переменных аффинно эквивалентна функции из класса Мэйорана—МакФарланда (см. далее теорему 16).

Класс $\mathfrak{B}_6$ содержит $5\,425\,430\,528 \simeq 2^{32,3}$ функций. Описание было дано С. В. Агиевичем [22] с использованием бент-квадратов, т. е. бент-прямоугольников при $r = k$ (см. теорему 10). Скажем, что две бент-функции *квадратно-эквивалентны*, если бент-квадрат одной из них может быть получен из бент-квадрата второй изменением знаков элементов и перестановкой строк и столбцов. Пусть $r = k = 3$. Все функции класса $\mathfrak{B}_6$ разбиваются на восемь классов квадратной эквивалентности. Ниже приводятся соответствующие бент-квадраты размера $2^3 \times 2^3$ и количество функций в каждом классе.

8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8	-4 4 4 4 0 0 0 0 4-4 4 4 0 0 0 0 4 4-4 4 0 0 0 0 4 4 4-4 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8	-4 4 4 4 0 0 0 0 4-4 4 4 0 0 0 0 0 0-4 4 4 4 0 0 0 0 4-4 4 4 0 0 4 4 0 0-4 4 0 0 4 4 0 0 4-4 0 0 0 0 0 0 0 0 8 0 0 0 0 0 0 0 0 8	-4 4 4 4 0 0 0 0 4-4 0 0 4 4 0 0 4 0-4 0 4 0 4 0 4 0 0-4 0 4 4 0 0 4 4 0 0 4-4 0 0 4 0 4-4 0 4 0 0 0 4 4 4-4 0 0 0 0 0 0 0 0 0 8
$(2^{15} \cdot 3^2 \cdot 5 \cdot 7)$	$(2^{18} \cdot 3 \cdot 7^2)$	$(2^{21} \cdot 3 \cdot 7^2)$	$(2^{25} \cdot 3 \cdot 7)$

-4 4 4 4 0 0 0 0 4-4 4 4 0 0 0 0 4 4-4 4 0 0 0 0 4 4 4-4 0 0 0 0 0 0 0 0-4 4 4 4 0 0 0 0 4-4 4 4 0 0 0 0 4 4-4 4 0 0 0 0 4 4 4-4	-4 4 4 4 0 0 0 0 4-4 4 4 0 0 0 0 0 0-4 4 4 4 0 0 0 0 4-4 4 4 0 0 0 0 0 0-4 4 4 4 0 0 0 0 4-4 4 4 4 4 0 0 0 0-4 4 4 4 0 0 0 0 4-4	-4 4 4 4 0 0 0 0 4-4 0 0 4 4 0 0 4 0-4 0 4 0 4 0 4 0 0-4 0 4 4 0 0 4 4 0-4 0 0 4 0 4 0 4 0-4 0 4 0 0 4 4 0 0-4 4 0 0 0 0 4 4 4-4	-6 2 2 2 2 2 2 2 2-6 2 2 2 2 2 2 2 2-6 2 2 2 2 2 2 2 2-6 2 2 2 2 2 2 2 2-6 2 2 2 2 2 2 2 2-6 2 2 2 2 2 2 2 2-6 2 2 2 2 2 2 2 2-6
$(2^{19} \cdot 7^2)$	$(2^{20} \cdot 3^2 \cdot 7^2)$	$(2^{23} \cdot 3 \cdot 7^2)$	$(2^{23} \cdot 3^2 \cdot 5 \cdot 7)$

Отметим, что мощность $\mathfrak{B}_6$ была найдена раньше в диссертации Б. Пренела [104]. В 2004 г. авторы [92] перечислили функции класса $\mathfrak{B}_6$ способом, отличным от приведенного в [22].

$n = 8$. Аффинная классификация бент-функций от восьми переменных степени не выше 3 была получена в работах [30, 74] (см. также работу [23], посвященную кубическим бент-функциям специального вида). Бент-функции от восьми переменных степени не выше 3 делятся на 10 классов аффинной эквивалентности, представителями которых являются:

$v_1v_2 \oplus v_3v_4 \oplus v_5v_6 \oplus v_7v_8$,
$v_1v_2v_3 \oplus v_1v_4 \oplus v_2v_5 \oplus v_3v_6 \oplus v_7v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4 \oplus v_2v_6 \oplus v_1v_7 \oplus v_5v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_1v_3 \oplus v_1v_5 \oplus v_2v_6 \oplus v_3v_4 \oplus v_7v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4v_6 \oplus v_3v_5 \oplus v_2v_6 \oplus v_2v_5 \oplus v_1v_7 \oplus v_4v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4v_6 \oplus v_3v_5 \oplus v_1v_3 \oplus v_1v_4 \oplus v_2v_7 \oplus v_6v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4v_6 \oplus v_3v_5 \oplus v_2v_6 \oplus v_2v_5 \oplus v_1v_2 \oplus v_1v_3 \oplus v_1v_4 \oplus v_7v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4v_6 \oplus v_3v_5 \oplus v_1v_6 \oplus v_2v_7 \oplus v_4v_8$,
$v_1v_2v_7 \oplus v_3v_4v_7 \oplus v_5v_6v_7 \oplus v_1v_4 \oplus v_3v_6 \oplus v_2v_5 \oplus v_4v_5 \oplus v_7v_8$,
$v_1v_2v_3 \oplus v_2v_4v_5 \oplus v_3v_4v_6 \oplus v_1v_4v_7 \oplus v_3v_5 \oplus v_2v_7 \oplus v_1v_5 \oplus v_1v_6 \oplus v_4v_8$.

В диссертации [30] также показано, что все эти функции аффинно эквивалентны функциям из класса Мэйорана—МакФарланда.

Нижняя $2^{70,4}$ и верхняя $2^{129,2}$ оценки числа всех функций в классе $\mathfrak{B}_8$ были получены соответственно в [22] и [86]. Некоторые результаты по частичному описанию класса $\mathfrak{B}_8$ на основе исследования групп автоморфизмов бент-функций приводит У. Демпвольф в работах [60, 61]. М. Янг, К. Менг и Х. Жанг [113] показали, что множество $\mathfrak{B}_8$ состоит не менее чем из 129 классов аффинной эквивалентности. Представители всех найденных ими классов приводятся в их работе. Это 53 функции вида $\mathrm{tr}(\alpha^i \boldsymbol{v}^{d_1} + \alpha^j \boldsymbol{v}^{d_2} + \alpha^k \boldsymbol{v}^{d_3})$ и 76 функций вида $\mathrm{tr}(\alpha^i \boldsymbol{v}^{d_1} + \alpha^j \boldsymbol{v}^{d_2} + \alpha^k \boldsymbol{v}^{d_3} + \alpha^\ell \boldsymbol{v}^{d_4})$, где $\mathrm{tr} : GF(2^8) \to GF(2)$ — функция следа. Авторы [72] показали, что множество $\mathfrak{B}_8 \cap \mathcal{PS}$ содержит функции из не менее чем шести классов аффинной эквивалентности, где $\mathcal{PS}$ — класс бент-функций, полученных методом частичного расщепления (см. далее теорему 18).

По последним данным (2009) аффинная классификация бент-функций от восьми переменных четвертой степени завершена [84]. Описаны все 536 возможных вариантов для части четвертой степени[2] АНФ бент-функции от восьми переменных. Установлено точное число всех бент-функций от восьми переменных [84]. Оно равно
$2^9 \times 193\,887\,869\,660\,028\,067\,003\,488\,010\,240 \simeq 2^{106,29}$.

При $n \geqslant 10$ класс $\mathfrak{B}_n$ не описан, его мощность неизвестна. В работе [113] построено большое число бент-функций от десяти переменных; установлено, что среди них содержится как минимум несколько сотен попарно аффинно неэквивалентных функций. Некоторую информацию о классах $\mathfrak{B}_{10}$, $\mathfrak{B}_{12}$ можно найти на сайте [61].

2.4. Оценки числа бент-функций

Информации об оценках числа бент-функций от n переменных немного. Приведем нижнюю оценку этого числа, которую дает конструкция Мэйорана—МакФарланда (см. далее теорему 16).

Теорема 11. Справедливо $|\mathfrak{B}_n| \geqslant 2^{2^{n/2}}(2^{n/2})!$.

[2] Под *частью степени i АНФ функции* понимаем набор всех тех слагаемых ее АНФ, степень которых равна i.

Асимптотически, эта оценка имеет вид $(\frac{2^{(n/2)+1}}{e})^{2^{n/2}}\sqrt{2^{(n/2)+1}\pi}$, или, если совсем грубо, $2^{2^{n/2}}$. Следует отметить, что в работе [22] приводится уточнение оценки теоремы 11, являющееся на данный момент лучшим. Однако охарактеризовать асимптотическое поведение оценки [22] достаточно трудно.

Тривиальная верхняя оценка следует из того факта, что, согласно теореме 2, степень бент-функции не превышает $n/2$. Имеем

$$|\mathfrak{B}_n| \leqslant 2^{1+\binom{n}{1}+\binom{n}{2}+\ldots+\binom{n}{n/2}} = 2^{2^{n-1}+\frac{1}{2}\binom{n}{n/2}}.$$

К. Карле и А. Клаппер в 2002 г. [48] немного улучшили эту оценку:

Теорема 12. Пусть $n \geqslant 6$ и выполняется $\varepsilon = \frac{1}{2^{O(\sqrt{2^n/n})}}$. Тогда

$$|\mathfrak{B}_n| \leqslant 2^{2^{n-1}+\frac{1}{2}\binom{n}{n/2}-2^{n/2}+(n/2)+1}(1+\varepsilon) + 2^{2^{n-1}-\frac{1}{2}\binom{n}{n/2}}.$$

Однако по-прежнему верхняя оценка близка к тривиальной 2^{2^n}. Верхняя оценка обсуждается также в работе [112].

Кажется интересным, что аналогичная проблема сильного разрыва между нижней и верхней оценками наблюдается и для числа других комбинаторных объектов.

Например, для совершенных двоичных кодов длины $n = 2^s - 1$ с расстоянием 3 (см. определение в [9]). Нижнюю оценку вида $2^{2^{n/2}}$ дает конструкция Ю. Л. Васильева 1962 г. [3], и с ней схожа, на мой взгляд, конструкция Мэйорана—МакФарланда для бент-функций; схожа по своей простоте и изяществу, и той роли основного, базового класса, которую играет в множестве бент-функций. А тип верхней оценки числа совершенных кодов по-прежнему остается тривиальным: 2^{2^n}. Небольшие улучшения нижней и верхней оценок приводятся соответственно в [82] и [1].

2.5. Конструкции бент-функций

Очень сложно не только классифицировать бент-функции, но и предложить отдельные способы их построения. В этом разделе мы следуем в основном работе [40] К. Карле, в которой всевозможные конструкции бент-функций представлены наиболее полно. Конструкции принято делить на *первичные* (primary) и *вторичные* (secondary). К первой группе относят те, с помощью которых бент-функции строятся напрямую, ко второй группе — конструкции, опирающиеся на уже известные бент-функции (например, от меньшего числа переменных).

Ко вторичным конструкциям относится простая *итеративная конструкция* [110].

Теорема 13. Функция $f(\boldsymbol{u'}, \boldsymbol{u''}) = g(\boldsymbol{u'}) \oplus h(\boldsymbol{u''})$, где векторы $\boldsymbol{u'}$, $\boldsymbol{u''}$ имеют четные длины r, k соответственно, является бент-функцией тогда и только тогда, когда функции g, h — бент-функции.

Конструкция легко может быть описана в терминах бент-прямоугольников [24]. Приведем следующее обобщение этой простой конструкции, полученное в [37].

Теорема 14. Пусть $n = r + k$, где r и k четны, f — булева функция от n переменных. Пусть $\boldsymbol{u'}$, $\boldsymbol{u''}$ пробегают $\mathbb{Z}_2^r$ и $\mathbb{Z}_2^k$ соответственно. Предположим, что функции

$$f_{\boldsymbol{u''}}(\boldsymbol{u'}) = f(\boldsymbol{u'}, \boldsymbol{u''})$$

являются бент-функциями при любых $\boldsymbol{u''}$. Определим $g_{\boldsymbol{u'}}(\boldsymbol{u''}) = \widetilde{f_{\boldsymbol{u''}}}(\boldsymbol{u'})$. Тогда f — бент-функция, если и только если $g_{\boldsymbol{u'}}$ — бент-функция для любого $\boldsymbol{u'}$.

Заметим, что теорема 13 следует из теоремы 14. Итеративный способ построения бент-функций от $n+2$ переменных из бент-функций от n переменных приводится в [56]. В качестве упражнения можно доказать следующий факт (см. [75]).

Теорема 15. Пусть f — булева функция от n переменных, h — перестановка на $\mathbb{Z}_2^n$. Обозначим через $h_1, \ldots, h_n$ булевы функции такие, что $h(\boldsymbol{v}) = (h_1(\boldsymbol{v}), \ldots, h_n(\boldsymbol{v}))$. Функция $f \circ h^{-1}$ является бент-функцией, если для каждого $\boldsymbol{u}$ выполняется

$$\operatorname{dist}(f, \bigoplus_{i=1}^{n} u_i h_i) = 2^{n-1} \pm 2^{(n/2)-1}.$$

К первичным конструкциям принадлежит простая и богатая *конструкция Мэйорана—МакФарланда* 1973 г. [63, 91].

Теорема 16. Пусть h — любая перестановка на $\mathbb{Z}_2^{n/2}$, пусть g — произвольная булева функция от $n/2$ переменных. Тогда функция $f(\boldsymbol{u'}, \boldsymbol{u''}) = \langle \boldsymbol{u'}, h(\boldsymbol{u''}) \rangle \oplus g(\boldsymbol{u''})$ является бент-функцией от n переменных.

Основной идеей конструкции служит, по выражению К. Карле [40], «конкатенация аффинных функций». Действительно, при каждом фиксированном значении переменных из второй половины функция f является аффинной от $n/2$ первых переменных. С другой стороны, аффинные функции возникают и при рассмотрении соответствующих бент-квадратов. А именно, бент-функция принадлежит классу Мэйорана—МакФарланда, если и только если строки и столбцы ее бент-квадрата являются спектральными векторами аффинных булевых функций [22].

Из теоремы легко следует, что существуют бент-функции с любой степенью нелинейности d, такой, что $2 \leqslant d \leqslant n/2$. Итак, в теореме 16 переменные функции f разбиваются пополам. В 2004 г. К. Карле [39] (см. также [40]) обобщил идею Мэйорана—МакФарланда, рассмотрев разбиение переменных на неравные части.

Теорема 17. Пусть $n = r + k$. Пусть $h : \mathbb{Z}_2^k \to \mathbb{Z}_2^r$ — любое отображение, такое, что для каждого вектора $\boldsymbol{u}$ длины r множество $h^{-1}(\boldsymbol{u})$ образует подпространство в $\mathbb{Z}_2^k$ размерности $n - 2r$. Пусть g — булева функция от k переменных, сужение которой на $h^{-1}(\boldsymbol{u})$ для каждого $\boldsymbol{u}$ является бент-функцией при $n > 2r$. Тогда булева функция $f(\boldsymbol{u'}, \boldsymbol{u''}) = \langle \boldsymbol{u'}, h(\boldsymbol{u''}) \rangle \oplus g(\boldsymbol{u''})$ является бент-функцией от n переменных.

Отметим, что конструкция К. Карле имеет сильные сходства с методом описания бент-функций, предложенным В. В. Ященко [19] еще в 1997 г. (см. выше теорему 7).

Теорема 16 представляет собой частный случай теоремы 17 при $r = k = n/2$.

Следующая первичная конструкция Дж. Диллона [63] 1974 г. опирается на специальные семейства подпространств n-мерного пространства и носит название *частичного расщепления* (Partial Spreads).

Пусть $\operatorname{Ind}_S : \mathbb{Z}_2^n \to \mathbb{Z}_2$ — характеристическая функция подмножества $S \subseteq \mathbb{Z}_2^n$.

Теорема 18. Пусть число q равно $2^{(n/2)-1}$ или $2^{(n/2)-1} + 1$. Пусть $L_1, \ldots, L_q$ — линейные подпространства размерности $n/2$ пространства $\mathbb{Z}_2^n$ такие, что любые два из них пересекаются лишь по нулевому вектору. Тогда функция $f(\boldsymbol{v}) = \bigoplus_{i=1}^{q} \operatorname{Ind}_{L_i}(\boldsymbol{v})$ является бент-функцией.

Случай $q = 2^{(n/2)-1}$ определяет класс бент-функций $\mathcal{PS}^-$.

Случай $q = 2^{(n/2)-1} + 1$ задает класс бент-функций $\mathcal{PS}^+$.

Вместе $\mathcal{PS}^-$ и $\mathcal{PS}^+$ составляют класс $\mathcal{PS}$.

Более общие геометрические конструкции можно найти, например, в работе [36].

Приведем несколько алгебраических конструкций.

Первая серия конструкций называется *степенные* или *мономиальные бент-функции* (power/monomial bent functions). Пусть векторное пространство $\mathbb{Z}_2^n$ отождествляется с полем Галуа $GF(2^n)$. Булевы функции от n переменных можно рассматривать как функции из $GF(2^n)$ в $GF(2)$, сопоставляя каждому вектору $\boldsymbol{v}$ соответствующий элемент поля $GF(2^n)$, который будем обозначать тем же символом. Пусть $\mathrm{tr} : GF(2^n) \to GF(2)$ — *функция следа*, т. е. $\mathrm{tr}(\boldsymbol{v}) = \boldsymbol{v} + \boldsymbol{v}^2 + \cdots + \boldsymbol{v}^{2^{n-1}}$. Бент-функции, имеющие вид

$$f(\boldsymbol{v}) = \mathrm{tr}(a\boldsymbol{v}^d),$$

где $a \in GF^*(2^n)$ — некоторый параметр, называются *степенными* или *мономиальными*, а целое число d называется *бент-показателем*. Здесь $GF^*(2^n)$ — множество ненулевых элементов поля. Бент-функции такого вида интересны в первую очередь для криптографических приложений в силу своей простой вычислимости. Хотя криптографы до сих пор не определились: считать простоту вычислимости бент-функции ее достоинством или скорее недостатком [40].

Пусть $\gcd(\cdot,\cdot)$ — наибольший общий делитель двух чисел.

Теорема 19. Следующие значения d являются бент-показателями:

$d = 2^{n/2} - 1$	(**Диллон ⋄, 1974, [63]**);
$d = 2^i + 1$, где $\frac{n}{\gcd(n,i)}$ четно	(**показатель Голда †**);
$d = 2^{2k} - 2^k + 1$, где $\gcd(k,n) = 1$	(**показатель Касами**);
$d = (2^k+1)^2$, где $n = 4k$, k нечетно	(**Канто–Леандер †, 2004, [87]**);
$d = 2^{2k} + 2^k + 1$, где $n = 6k$	(**Канто–Шарпин–Карегян †, 2006, [35]**).

Известно, что три типа степенных бент-функций (в теореме их показатели помечены знаком †) можно описать с помощью конструкции Мэйорана—МакФарланда, а один тип (помечен знаком ⋄) содержится в классе $\mathcal{PS}^-$. Существуют ли степенные бент-функции с другими показателями? Можно ли для степенных бент-функций найти простое комбинаторное описание? Ответов на эти вопросы пока нет.

Вторая серия бент-функций состоит из функций вида

$$f(\boldsymbol{v}) = \mathrm{tr}(a_1\boldsymbol{v}^{d_1} + a_2\boldsymbol{v}^{d_2}) \tag{3}$$

для подходящих элементов $a_1, a_2 \in GF(2^n)$ и показателей d_1, d_2. Известны примеры таких функций со специальными степенными показателями — так называемыми *показателями Нихо* вида $d \equiv 2^i \bmod (2^{n/2} - 1)$. Без ограничения общности [67] пусть первый показатель равен $d_1 = (2^{(n/2)} - 1)\frac{1}{2} + 1$. Справедлива [69]

Теорема 20. Если выполняется $d_2 = (2^{(n/2)} - 1)\lambda + 1$, где λ равно $1/6$, $1/4$ или 3, то существуют элементы $a_1, a_2 \in GF(2^n)$ такие, что (3) является бент-функцией.

Алгоритмические вопросы построения функций такого вида разбираются в [113].

Бент-функции вида $f(\boldsymbol{v}) = \sum\limits_{i=1}^{(n-1)/2} c_i\,\mathrm{tr}(\boldsymbol{v}^{1+2^i})$ изучались в [51, 76, 80, 81, 113, 115].

Следует отметить, что алгебраические конструкции бент-функций носят весьма случайный характер: каждый раз исследуются функции лишь некого специального вида. Общий алгебраический подход к описанию бент-функций мог бы основываться на том, что любая булева функция $f : GF(2^n) \to GF(2)$ может быть представлена

с помощью следа (в так называемой trace form), т. е. в виде

$$f(\boldsymbol{v}) = \mathrm{tr}\left(\sum_{d\in CS} a_d \boldsymbol{v}^d\right) = \sum_{d\in CS} \mathrm{tr}(a_d \boldsymbol{v}^d) \tag{4}$$

для подходящих элементов $a_d \in GF(2^n)$, где CS — множество представителей циклотомических классов по модулю $2^n - 1$. Эволюционный алгоритм на основе такого представления был предложен М. Янгом, К. Менгом и Х. Жангом [113]. Эта работа уже упоминалась нами в связи с классификацией бент-функций от 6 и 8 переменных. На основе многочисленных компьютерных исследований авторы делают в этой работе некоторые предположения относительно общего алгебраического вида бент-функций. В частности, они предполагают, что бент-функцию — представителя класса аффинной эквивалентности — можно представить в виде (4) с участием небольшого числа мономов. Причем более вероятными ненулевыми коэффициентами a_d в этом представлении авторы [113] считают те, для которых d является бент-показателем (см. теорему 19).

Но общего подхода к алгебраическому описанию бент-функций пока нет.

Более полно (с доказательствами) конструкции бент-функций представлены в обзорах [8, 40, 67], см. также другие конструкции в работах [37, 64].

2.6. Генерация бент-функций

Серия работ посвящена алгоритмическим методам построения бент-функций. Каждый метод основывается, как правило, на одном из возможных представлений булевой функции и использует те его особенности, которые проявляются в случае, когда булева функция оказывается бент-функцией. К таким базовым представлениям относятся: таблица истинности [92, 93], АНФ [70, 71], спектральный вектор булевой функции [55], представление с помощью следа [113] и др.

В диссертации Дж. Е. Фаллер [70] подробно разбираются эвристические методы построения бент-функций. Их основная идея заключается в постепенном изменении начальной булевой функции с улучшением тех или иных ее криптографических свойств, включающих нелинейность. Так, для построения бент-функций применяются: генетический алгоритм [94], алгоритмы случайного поиска [95, 70], алгоритм имитации отжига [54]. В диссертации [70] предлагается достаточно быстрый алгоритм построения псевдослучайных бент-функций степени не выше некоторой заданной: функции строятся из случайной квадратичной бент-функции g путем итеративного добавления к АНФ(g) слагаемых более высоких степеней. При этом основная трудность — «отбраковка» бóльшей части слагаемых — преодолевается за счет существенного использования комбинаторных свойств бент-функций.

В 2004 г. К. Менг с соавторами предложили алгоритм [92], позволяющий (теоретически) построить все бент-функции от любого числа переменных n. По сравнению с полным перебором сложность данного алгоритма ниже за счет использования соотношений между отдельными коэффициентами Уолша—Адамара произвольной булевой функции и спектральными векторами ее подфункций, а также за счет оперирования свойствами бент-функций, приведенными в теореме 4. Практически, данный алгоритм и его модификации оказались применимы пока только для генерации всех бент-функций от 6 переменных, всех однородных [105] бент-функций степени 3 от 8 переменных и доказательства несуществования однородных бент-функций степени 4 от 10 переменных.

С. В. Агиевичем [22] приводится алгоритм порождения достаточно большого числа бент-функций, основанный на использовании бент-прямоугольников (см. теорему 10). Данный алгоритм позволил установить лучшую на данный момент нижнюю оценку числа бент-функций от n переменных. Эволюционный алгоритм на основе представления булевых функций с помощью следа предложен в [113]. Подробнее о применении эволюционных вычислений для генерации бент-функций см. также в [55, 71, 93]. Отдельные аспекты порождения случайных бент-функций обсуждаются в работе [73].

2.7. Другие результаты

В 1998 г. Д. Оледжар и М. Станек [98] исследовали криптографические свойства случайной булевой функции от n переменных. В частности, ими была доказана

Теорема 21. Существует константа c такая, что при достаточно больших n почти для каждой булевой функции f от n переменных выполняется $N_f \geqslant 2^{n-1} - c\sqrt{n}\,2^{n/2}$.

Позднее в 2002 г. этот факт был независимо получен К. Карле [38].

Выражение «почти для каждой» следует понимать как «с вероятностью, стремящейся к 1».

Пусть нелинейность произвольной булевой функции g от n переменных имеет вид $N_g = 2^{n-1} - S(g)$, где $S(g)$ — некоторая функция. В 2006 г. Ф. Родье [107] установил асимптотическое значение нелинейности булевой функции. Пусть V^∞ — множество бесконечных двоичных последовательностей, почти все элементы которых равны нулю. Пусть $f : V^\infty \to \mathbb{Z}_2$. Обозначим через f_n сужение функции f на множество $\mathbb{Z}_2^n$ (см. подробнее [107]).

Теорема 22. Почти для каждой функции $f : V^\infty \to \mathbb{Z}_2$ верно

$$\lim_{n\to\infty} \frac{S(f_n)}{2^{n/2}\sqrt{n}} = \sqrt{2\ln 2}.$$

Т. е. с ростом n нелинейность случайной булевой функции от n переменных становится достаточно высокой, и даже сопоставимой с нелинейностью бент-функции!

Для криптографических приложений булева функция кроме нелинейности должна обладать целым рядом других свойств. Можно сказать, что теорема 22 «гарантирует» возможность выбора функции с высокой нелинейностью не в ущерб этим свойствам. Но хотя нелинейность почти всех булевых функций высока, это не означает, что такие функции легко построить. Подобные «парадоксы» уже возникали для булевых функций, например при исследовании их сложностных характеристик[3]. В данном случае асимптотическая оценка теоремы 22 задает некий *уровень* нелинейности, с которым имеет смысл сравнивать нелинейность той или иной криптографической булевой функции [108].

В 2006 г. У. Демпвольф [60] предпринял попытку исследования групп автоморфизмов бент-функций. Более точно — групп автоморфизмов соответствующих элементарных адамаровых разностных множеств (см. теорему 5). У. Демпвольф показал, что каждое такое разностное множество, при наличии определенного свойства у его группы автоморфизмов, относится к одному из пяти указанных им специальных классов.

В целом группы автоморфизмов бент-функций исследованы пока крайне мало.

[3] К. Шенноном было доказано, что почти все булевы функции имеют очень большую сложность реализации, асимптотически равную сложности «самой сложной» функции [11], но ни одну такую функцию построить пока не удалось.

2.8. Векторные бент-функции

С 90-х годов XX века стали исследоваться функции $f : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$, получившие название *векторных булевых функций*, или (n, m)*-функций*. Интерес к ним вызван тем, что нелинейные такие функции имеют непосредственные криптографические приложения. Например, в шифрах они используются в качестве S-блоков.

Рассмотрим нелинейные свойства векторных функций.

Преобразованием Уолша—Адамара (n, m)-функции f называется отображение $W_f^{\mathrm{vect}} : \mathbb{Z}_2^n \times \mathbb{Z}_2^m \to \mathbb{Z}$, заданное равенством

$$W_f^{\mathrm{vect}}(\boldsymbol{a}, \boldsymbol{b}) = \sum_{\boldsymbol{v} \in \mathbb{Z}_2^n} (-1)^{\langle \boldsymbol{a}, \boldsymbol{v} \rangle \oplus \langle \boldsymbol{b}, f(\boldsymbol{v}) \rangle} \text{ для любых } \boldsymbol{a} \in \mathbb{Z}_2^n, \boldsymbol{b} \in \mathbb{Z}_2^m.$$

Нелинейностью (n, m)-функции f называется минимальная из нелинейностей булевых функций $f_{\boldsymbol{b}}$ от n переменных, где $f_{\boldsymbol{b}}(\boldsymbol{v}) = \langle \boldsymbol{b}, f(\boldsymbol{v}) \rangle$ при различных значениях $\boldsymbol{b} \in \mathbb{Z}_2^m$, $\boldsymbol{b} \neq \boldsymbol{0}$. Справедливо

$$N_f = \min_{\boldsymbol{b} \in (\mathbb{Z}_2^m)^*} \operatorname{dist}(f_{\boldsymbol{b}}, \mathfrak{A}_n) = 2^{n-1} - \frac{1}{2} \max_{\boldsymbol{a} \in \mathbb{Z}_2^n, \boldsymbol{b} \in (\mathbb{Z}_2^m)^*} |W_f^{\mathrm{vect}}(\boldsymbol{a}, \boldsymbol{b})|.$$

Здесь через $(\mathbb{Z}_2^m)^*$ обозначено множество ненулевых двоичных векторов длины m. Для нелинейности векторной булевой функции имеется та же самая верхняя оценка, что и в случае обычной булевой функции:

$$N_f \leqslant 2^{n-1} - 2^{(n/2)-1}. \tag{5}$$

Векторная (n, m)-функция называется *бент-функцией*, если параметр N_f достигает своего максимального возможного значения, т. е. если каждая булева функция $f_{\boldsymbol{b}}$, где $\boldsymbol{b} \in (\mathbb{Z}_2^m)^*$, является бент-функцией. Справедлива

Теорема 23. Векторная (n, m)-функция f является бент-функцией тогда и только тогда, когда для любого ненулевого вектора $\boldsymbol{u}$ функция $f(\boldsymbol{v}) \oplus f(\boldsymbol{v} \oplus \boldsymbol{u})$ сбалансирована, т. е. принимает каждое из 2^m возможных значений ровно 2^{n-m} раз.

Следующий важный факт о существовании векторных бент-функций получила К. Ньюберг [96] в 1991 г.

Теорема 24. Бент (n, m)-функции существуют тогда и только тогда, когда n четно и $m \leqslant n/2$.

Легко построить примеры таких функций, например, применяя конструкцию Мэйорана—МакФарланда в новой, векторной, форме, предложенной К. Ньюберг [96]. Отождествим пространство $\mathbb{Z}_2^{n/2}$ с полем Галуа $GF(2^{n/2})$, а пространство $\mathbb{Z}_2^n$ — с прямым произведением $GF(2^{n/2}) \times GF(2^{n/2})$. Пусть n четно, $m \leqslant n/2$. Справедлива

Теорема 25. Пусть $h : GF(2^{n/2}) \to GF(2^{n/2})$ — любое взаимно однозначное отображение, g — произвольная $(n/2, m)$-функция. Пусть $L : GF(2^{n/2}) \to \mathbb{Z}_2^{n/2}$ — любое линейное или аффинное отображение «на». Тогда векторная (n, m)-функция $f(\boldsymbol{u'}, \boldsymbol{u''}) = L(\boldsymbol{u'} \cdot h(\boldsymbol{u''})) \oplus g(\boldsymbol{u''})$ является бент-функцией.

Конструкция Мэйорана—МакФарланда является не единственной, которая переносится на векторный случай (см. подробнее [41]).

Поскольку бент (n, m)-функций не существует при $m > n/2$, то оценка (5) в этом случае не точна. В 1971 г. В. М. Сидельников [12] и независимо в 1994 г. Ф. Шабат, С. Ваденай [50] установили следующий факт.

Теорема 26. Пусть $m \geqslant n-1$. Для любой (n,m)-функции f выполняется

$$N_f \leqslant 2^{n-1} - \frac{1}{2}\sqrt{3(2^n) - 2 - 2\frac{(2^n-1)(2^{n-1}-1)}{2^m-1}}. \tag{6}$$

При $n/2 < m < n-1$ оценки, улучшающей (5), пока не известно.

Случай $n = m$ выделяется особо. В этом случае оценка (6) имеет вид

$$N_f \leqslant 2^{n-1} - 2^{(n-1)/2}.$$

Векторная (n,n)-функция f называется *почти бент-функцией* (AB function — almost bent function), если параметр N_f достигает своего максимального возможного значения, $N_f = 2^{n-1} - 2^{(n-1)/2}$. Следует отметить, что по смыслу слово «почти» здесь совершенно лишнее, поскольку речь идет о максимальном значении N_f. Но термин в таком виде уже устоялся. AB-функции существуют, только если n нечетно. К. Карле, П. Шарпин и В. Зиновьев [42] доказали, что степень нелинейности любой такой функции не превышает величины $(n+1)/2$.

Более широким является класс APN-функций.

Эти векторные (n,n)-функции стала рассматривать в 1993 г. К. Ньюберг [97] при исследовании устойчивости шифров к дифференциальному криптоанализу [28]. Стойкость S-блока, заданного векторной функцией f, к дифференциальному криптоанализу тем выше, чем меньше значение $\delta_f = \max\limits_{\boldsymbol{a}\in(\mathbb{Z}_2^n)^*, \boldsymbol{b}\in\mathbb{Z}_2^n} \delta_{f,\boldsymbol{a},\boldsymbol{b}}$, где через $\delta_{f,\boldsymbol{a},\boldsymbol{b}}$ обозначено число решений уравнения $f(\boldsymbol{v}) \oplus f(\boldsymbol{v} \oplus \boldsymbol{a}) = \boldsymbol{b}$. Параметр δ_f и его связи с другими нелинейными характеристиками исследовались в работах [7, 20, 33]. Наименьшее возможное значение параметра δ_f равно двум[4]. Векторная (n,n)-функция, для которой этот минимум достигается, называется *почти совершенно нелинейной* (APN function — almost perfectly nonlinear function). И снова, по иронии, слово «почти» здесь абсолютно ни при чем. Эквивалентно, APN-функция может быть определена как функция, сужение которой на любое двумерное аффинное подпространство пространства $\mathbb{Z}_2^n$ является неаффинной функцией. При нечетном n APN-функции существуют. А вот существуют ли они при четном n? — пока открытый вопрос.

AB- и APN- функции тесно связаны (см. обзор результатов в [41]).

Теорема 27. Каждая AB-функция является APN-функцией.

Теорема 28. Квадратичная APN-функция является AB-функцией.

Приведем одно определение для обычных булевых функций. Булева функция f называется *платовидной* (plateaued function), если существует положительное целое число M такое, что любой коэффициент Уолша—Адамара $W_f(\boldsymbol{v})$ равен 0 или $\pm M$. Из равенства Парсеваля следует, что $M = 2^\beta$, и показатель β может принимать целые значения от $n/2$ до n. Число $2(n-\beta)$ называют *порядком* платовидной функции f. Бент-функции и аффинные функции являются крайними частными случаями платовидных функций (порядков m и 0 соответственно). Справедлива

Теорема 29. Векторная функция f является AB-функцией тогда и только тогда, когда она APN-функция и все булевы функции $f_{\boldsymbol{b}}$ при $\boldsymbol{b} \neq \boldsymbol{0}$ являются платовидными, причем одного порядка.

[4]Интересно, что при рассмотрении q-значных векторных функций, $q \neq 2$, возможно и $\delta_f = 1$.

Более общим понятием по отношению к понятию APN-функции является следующее. Векторная (n, n)-функция f называется *дифференциально δ-равномерной* (differential δ-uniform), δ — целое число, если уравнение $f(\boldsymbol{v}) \oplus f(\boldsymbol{v} \oplus \boldsymbol{a}) = \boldsymbol{b}$ при любых $\boldsymbol{a} \in (\mathbb{Z}_2^n)^*$, $\boldsymbol{b} \in \mathbb{Z}_2^n$ имеет не более δ решений, т. е., другими словами, $\delta_f = \delta$. APN-функции представляют собой частный случай таких функций при $\delta = 2$. Дифференциально 4-равномерные функции (см., например, [29]) используются в S-блоках симметричного алгоритма блочного шифрования AES (или Rijndael), являющегося с 26 мая 2002 г. американским стандартом шифрования.

AB, APN, δ-равномерные функции и вопросы их эквивалентности широко исследуются. В частности [32], уже выдвинута гипотеза, что все степенные AB- и APN-функции найдены (Х. Доббертин [65]) и обозначена проблема существования новых комбинаторных конструкций таких функций (см. подробнее [31, 41]). При $n \leqslant 25$ для APN-функций и при $n \leqslant 33$ для AB-функций гипотеза Доббертина уже подтвердилась [66, 88].

За пределами обзора остались *скрюченные функции* (crooked functions) — специальный подкласс APN-функций, введенный в 1998 г. Т. Бендингом и Д. Г. Фон-дер-Флаассом [25]. С помощью таких функций оказалось возможно строить новые дистанционно регулярные графы, симметричные схемы отношений и равномерно упакованные коды типа БЧХ и Препараты [57, 58] (см. также на эту тему работу [34]).

В данном обзоре остались незатронутыми и многие другие интересные темы.

Выражаю свою искреннюю признательность С. В. Агиевичу (Минск, Белоруссия) и Франсуа Родье (Марсель, Франция) за ценные замечания и полезные обсуждения. С большим удовольствием благодарю Лилию Будагян из университета Бергена (Норвегия) за консультации по векторным бент-функциям.

ЛИТЕРАТУРА

1. *Августинович С. В.* Об одном свойстве совершенных двоичных кодов // Дискрет. анализ и исслед. операций. 1995. Т. 2. № 1. С. 4–6.
2. *Амбросимов А. С.* Свойства бент-функций q-значной логики над конечными полями // Дискретная математика. 1994. Т. 6. № 3. С. 50–60.
3. *Васильев Ю. Л.* О негрупповых плотно упакованных кодах // Проблемы кибернетики. 1962. Вып. 8. С. 337–339.
4. *Кузнецов Ю. В., Шкарин С. А.* Коды Рида—Маллера (обзор публикаций) // Математические вопросы кибернетики. 1996. Вып. 6. С. 5–50.
5. *Кузьмин А. С., Марков В. Т., Нечаев А. А. и др.* Бент-функции и гипербент-функции над полем из 2^l элементов // Проблемы передачи информации. 2008. Т. 44. Вып. 1. С. 15–37.
6. *Логачев О. А., Сальников А. А., Ященко В. В.* Бент-функции на конечной абелевой группе // Дискретная математика. 1997. Т. 9. № 4. С. 3–20.
7. *Логачев О. А., Сальников А. А., Ященко В. В.* Некоторые характеристики «нелинейности» групповых отображений // Дискрет. анализ и исслед. операций. Сер. 1. 2001. Т. 8. № 1. С. 40–54.
8. *Логачев О. А., Сальников А. А., Ященко В. В.* Булевы функции в теории кодирования и криптологии. М.: Московский центр непрерывного математического образования, 2004. 470 с.
9. *Мак-Вильямс Ф. Дж., Слоэн Н. Дж. А.* Теория кодов, исправляющих ошибки. М.: Связь, 1979. 745 с.

10. *Молдовян А. А., Молдовян Н. А., Еремеев М. А.* Криптография: от примитивов к синтезу алгоритмов. СПб.: БХВ-Петербург, 2004.
11. *Нигматуллин Р. Г.* Сложность булевых функций. М.: Наука, 1991. 240 с.
12. *Сидельников В. М.* О взаимной корреляции последовательностей // Проблемы кибернетики. 1971. Т. 24. С. 15–42.
13. *Сидельников В. М.* Об экстремальных многочленах, используемых при оценках мощности кода // Проблемы передачи информации. 1980. Т. 14. Вып. 3. С. 17–30.
14. *Солодовников В. И.* Бент-функции из конечной абелевой группы в конечную абелеву группу // Дискретная математика. 2002. Т. 14. № 1. С. 99–113.
15. *Токарева Н. Н.* Бент-функции с более сильными свойствами нелинейности: k-бент-функции // Дискрет. анализ и исслед. операций. Сер. 1. 2007. Т. 14. № 4. С. 76–102.
16. *Токарева Н. Н.* Обобщения бент-функций. Обзор // Дискрет. анализ и исслед. операций. 2009. Т. 16 (готовится к печати). Доступен на `www.math.nsc.ru/~tokareva`.
17. *Холл М.* Комбинаторика. М.: Мир, 1970. 424 с.
18. *Черемушкин А. В.* Методы аффинной и линейной классификации булевых функций // Труды по дискретной математике. М.: Физматлит, 2001. Т. 4. С. 273–314.
19. *Ященко В. В.* О критерии распространения для булевых функций и о бент-функциях // Пробл. передачи информации. 1997. Т. 33. Вып. 1. С. 75–86.
20. *Ященко В. В.* О двух характеристиках нелинейности булевых отображений // Дискрет. анализ и исслед. операций. Сер. 1. 1998. Т. 5. № 2. С. 90–96.
21. *Adams C.* On immunity against Biham and Shamir's «differential cryptanalysis» // Information Processing Letters. 1992. V. 41. P. 77–80.
22. *Agievich S. V.* On the representation of bent functions by bent rectangles // Fifth Int. Petrozavodsk conf. on probabilistic methods in discrete mathematics (Petrozavodsk, Russia, June 1–6, 2000). Proc. Boston: VSP, 2000. P. 121–135. Available at `http://arxiv.org/abs/math/0502087`.
23. *Agievich S. V.* On the affine classification of cubic bent functions // Cryptology ePrint Archive, Report 2005/044, available at `http://eprint.iacr.org/`.
24. *Agievich S. V.* Bent rectangles // NATO Advanced Study Institute on Boolean Functions in Cryptology and Information Security (Zvenigorod, Russia. September 8–18, 2007). Proc: Netherlands, IOS Press, 2008. P. 3–22. Available at `http://arxiv.org/abs/0804.0209`.
25. *Bending T. D., Fon-Der-Flaass D. G.* Crooked Functions, Bent Functions and Distance Regular Graphs // Electronic J. Combinatorics. 1998. No. 5 (R34).
26. *Bernasconi A., Codenotti B.* Spectral analysis of Boolean functions as a graph eigenvalue problem // IEEE Trans. Computers. 1999. V. 48. No. 3. P. 345–351.
27. *Bernasconi A., Codenotti B., VanderKam J. M.* A characterization of bent functions in terms of strongly regular graphs // IEEE Trans. Computers. 2001. V. 50. No. 9. P. 984–985.
28. *Biham E., Shamir A.* Differential cryptanalysis of DES-like cryptosystems // J. Cryptology. 1991. V. 4. No. 1. P. 3–72.
29. *Bracken C., Leander G.* New families of functions with differential uniformity of 4 // Fourth International Conference BFCA — Boolean Functions: Cryptography and Applications (Copenhagen, Denmark, May 19–21, 2008). Proc. to appear. P. 190–194.
30. *Braeken A.* Cryptographic properties of Boolean functions and S-boxes // Ph. D. Thesis, Katholieke Univ. Leuven, 2006. Available at `http://www.cosic.esat.kuleuven.be/publications/thesis-129.pdf`.

31. *Budaghyan L., Carlet C., Leander G.* On inequivalence between known power APN functions // Fourth International Conference BFCA — Boolean Functions: Cryptography and Applications (Copenhagen, Denmark, May 19–21, 2008). Proc. to appear. P. 3–15.
32. *Budaghyan L.* Private Communication. 2008.
33. *Budaghyan L., Pott A.* On differential uniformity and nonlinearity of functions // Discrete Mathematics. 2009. V. 309. No. 1. P. 371–384.
34. *Byrne E., McGuire G.* On the non-existence of crooked functions on finite fields // WCC — International Workshop on Coding and Cryptography (Bergen, Norway, March 14–18, 2005). Proc. 2005. P. 316–324.
35. *Canteaut A., Charpin P., Kuyreghyan G.* A new class of monomial bent functions // Finite Fields and Applications. 2008. V. 14. No. 1. P. 221–241.
36. *Carlet C.* Generalized Partial Spreads // IEEE Trans. Inform. Theory. 1995. V. 41. No. 5. P. 1482–1487.
37. *Carlet C.* A construction of bent functions // Finite Fields and Applications, London mathematical society. 1996. Lecture series 233. P. 47–58.
38. *Carlet C.* On cryptographic complexity of Boolean functions // Proc. of the Sixth Conference on Finite Fields with Applications to Coding Theory, Cryptography and Related Areas. Springer, G. L. Mullen, H. Stichtenoth and H. Tapia-Recillas Eds. 2002. P. 53–69.
39. *Carlet C.* On the confusion and diffusion properties of Maiorana—McFarland's and extended Maiorana—McFarland's functions // Special Issue «Complexity Issues in Coding Theory and Cryptography» dedicated to Prof. Harald Niederreiter on the occasion of his 60th birthday, J. Complexity. 2004. V. 20. P. 182–204.
40. *Carlet C.* Boolean Functions for Cryptography and Error Correcting Codes // Chapter of the monograph «Boolean Methods and Models», Cambridge Univ. Press (P. Hammer, Y. Crama eds.), to appear. Prelim. version is available at `www-rocq.inria.fr/secret/Claude.Carlet/chap-fcts-Bool.pdf`.
41. *Carlet C.* Vectorial Boolean Functions for Cryptography // Chapter of the monograph «Boolean Methods and Models», Cambridge Univ. Press (P. Hammer, Y. Crama eds.), to appear. Prelim. version is available at `www-rocq.inria.fr/secret/Claude.Carlet/chap-vectorial-fcts.pdf`.
42. *Carlet C., Charpin P., Zinoviev V.* Codes, bent functions and permutations suitable for DES-like cryptosystems // Designs, Codes and Cryptography. 1998. V. 15. No. 2. P. 125–156.
43. *Carlet C., Danielsen L.-E., Parker M. G., Solé P.* Self Dual Bent Functions // Fourth International Conference BFCA — Boolean Functions: Cryptography and Applications (Copenhagen, Denmark, May 19–21, 2008). Proc. to appear. P. 39–52.
44. *Carlet C., Ding C.* Highly nonlinear mappings // J. Complexity. 2004. V. 20. No. 2–3. P. 205–244.
45. *Carlet C., Ding C., Niederreiter H.* Authentication schemes from highly nonlinear functions // Designs, Codes and Cryptography. 2006. V. 40. No. 1. P. 71–79.
46. *Carlet C., Guillot P.* A characterization of binary bent functions // J. Combin. Theory. Ser. A. 1996. V. 76. No. 2. P. 328–335.
47. *Carlet C., Guillot P.* An alternate characterization of the bentness of binary functions, with uniqueness // Designs, Codes and Cryptography. 1998. V. 14. P. 133–140.
48. *Carlet C., Klapper A.* Upper bounds on the numbers of resilient functions and of bent functions // 23rd Symposium on Information Theory (Benelux, Belgium. May, 2002).

Proc. 2002. P. 307–314. The full version will appear in Lecture Notes dedicated to Philippe Delsarte. Available at http://www.cs.engr.uky.edu/∼klapper/ps/bent.ps.

49. http://www.faqs.org/rfcs/rfc2144.html — CAST-128. Rfc 2144 — the cast-128 encryption algorithm— 1997.
50. *Chabaud F., Vaudenay S.* Links between Differential and Linear Cryptanalysis // Advances in Cryptology — EUROCRYPT '94, International Conference on the Theory and Application of Cryptographic Techniques. (Perugia, Italy. May 9–12, 1994) Proc. Springer, 1995. P. 356–365 (Lecture Notes in Comput. Sci. V. 950).
51. *Charpin P., Pasalic E., Tavernier C.* On bent and semi-bent quadratic Boolean functions // IEEE Trans. Inform. Theory. 2005. V. 51. No. 12. P. 4286–4298.
52. *Chase P. J., Dillon J. F., Lerche K. D.* Bent functions and difference sets // NSA R41 Technical Paper. September, 1970.
53. *Chee S., Lee S., Kim K.* Semi-bent Functions // Advances in Cryptology — ASIACRYPT '94 — 4th International Conference on the Theory and Applications of Cryptology. (Wollongong, Australia. November 28 – December 1, 1994). Proc. Berlin: Springer, 1995. P. 107–118 (Lecture Notes in Comput. Sci. V. 917).
54. *Clark J. A., Jacob J. L.* Two-stage optimisation in the design of Boolean functions // 5th Australian Conference on Information Security and Privacy. (Brisbane, Australia, July 10-12, 2000). Proc. Springer-Verlag, 2000. P. 242–254 (Lecture Notes in Comput. Sci. V. 1841).
55. *Clark J. A., Jacob J. L., Maitra S., Stanica P.* Almost Boolean Functions: the Design of Boolean Functions by Spectral Inversion. // Computational Intelligence. Special Issue on Evolutionary Computing in Cryptography and Security. 2004. V. 20. No. 3. P. 450–462.
56. *Climent J.-J., Garcia F. J., Requena V.* On the construction of bent functions of $n + 2$ variables from bent functions of n variables. // Advances in Math. of Communications. 2008. V. 2. No. 4. P. 421–431.
57. *Van Dam E. R., Fon-Der-Flaass D. G.* Uniformly Packed Codes and More Distance Regular Graphs from Crooked Functions // J. Algebraic Combinatorics. 2000. V. 12. No. 2. P. 115–121.
58. *Van Dam E. R., Fon-Der-Flaass D. G.* Codes, graphs, and schemes from nonlinear functions // European J. Combinatorics, 2003. V. 24. No. 1. P. 85–98.
59. *Delsarte P.* An algebraic approach to the association schemes of coding theory // Ph. D. Thesis, Univ. Catholique de Louvain, 1973.
60. *Dempwolff U.* Automorphisms and equivalence of bent functions and of difference sets in elementary Abelian 2-groups // Communications in Algebra. 2006. V. 34. No. 3. P. 1077–1131.
61. http://www.mathematik.uni-kl.de/~dempw/ — Homepage of U. Dempwolff. See the section «Bent Functions in Dimensions 8,10,12». 2009.
62. *Dillon J. F.* A survey of bent functions // The NSA Technical J. 1972. Special Issue. P. 191–215.
63. *Dillon J. F.* Elementary Hadamard Difference sets // Ph. D. Thesis. Univ. of Maryland, 1974.
64. *Dobbertin H.* Construction of bent functions and balanced Boolean functions with high nonlinearity // Fast Software Encryption, Second International Workshop — FSE'95. (Leuven, Belgium, December 14–16, 1994) Proc. Berlin: Springer, 1995. P. 61–74 (Lecture Notes in Comput. Sci. V. 1008).
65. *Dobbertin H.* Almost perfect nonlinear power functions over $GF(2^n)$: the Niho case // Inform. and Comput. 1999. V. 151. No. 1–2. P. 57–72.

66. *Dobbertin H.* Almost perfect nonlinear power functions over $GF(2^n)$: a new case for n divisible by 5 // Finite Fields and Applications FQ5 (Augsburg, Germany, August 2–6, 2000). Proc. Springer / Eds. D. Jungnickel, H. Niederreiter. 2000. P. 113–121.

67. *Dobbertin H., Leander G.* A survey of some recent results on bent functions // Sequences and their applications. – SETA 2004. Third Int. conference (Seoul, Korea, October 24–28, 2004). Revised selected papers. Berlin: Springer, 2005. P. 1–29 (Lecture Notes in Comput. Sci. V. 3486).

68. *Dobbertin H., Leander G.* Cryptographer's Toolkit for Construction of 8-Bit Bent Functions // Cryptology ePrint Archive, Report 2005/089, available at `http://eprint.iacr.org/`.

69. *Dobbertin H., Leander G., Canteaut A., et al.* Construction of Bent Functions via Niho Power Functions // J. Combin. Theory. Ser. A. 2006. V. 113. No. 5. P. 779–798. Available at `http://www-rocq.inria.fr/secret/Anne.Canteaut/Publications/index-pub.html`.

70. *Fuller J. E.* Analysis of affine equivalent Boolean functions for cryptography // Ph. D. thesis, Queensland University of Technology. Brisbane, Australia. 2003. Available at `http://eprints.qut.edu.au/15828/`.

71. *Fuller J. E., Dawson E., Millan W.* Evolutionary generation of bent functions for cryptography // The 2003 Congress on Evolutionary Computation. 2003. CEC apos;03. V. 3. P. 1655–1661.

72. *Gangopadhyay S., Sharma D., Sarkar S., Maitra S.* On Affine (Non) Equivalence of Bent Functions // CECC'08 — Central European Conference on Cryptography (Graz, Austria, July 2–4, 2008). Proc. 2008. Available at `http://www.math.tugraz.at/~cecc08/abstracts/cecc08_abstract_25.pdf`.

73. *Grocholewska-Czuryło A.* A study of differences between bent functions constructed using Rothaus method and randomly generated bent functions // J. Telecommunications and Information Technology. 2003. No. 4. P. 19–24. Available at `http://www.itl.waw.pl/czasopisma/JTIT/2003/4/19.pdf`.

74. *Hou X.-D.* Cubic bent functions // Discrete Mathematics. 1998. V. 189. P. 149–161.

75. *Hou X.-D., Langevin P.* Results on bent functions // J. Comb. Theory, Series A. 1997. V. 80. P. 232–246.

76. *Hu H., Feng D.* On quadratic bent functions in polynomial forms // IEEE Trans. Inform. Theory 2007. V. 53. No. 7. P. 2610–2615.

77. *Kantor W. M.* Codes, Quadratic Forms and Finite Geometries // Proceedings of Symposia in Applied Math. 1995. V. 50. P. 153–177. Available at `http://darkwing.uoregon.edu/~kantor/`.

78. *Kavut S., Maitra S., Yucel M. D.* Search for Boolean functions with excellent profiles in the rotation symmetric class // IEEE Trans. Inform. Theory. 2007. V. 53. No. 5. P. 1743–1751.

79. *Kerdock A. M.* A class of low-rate non-linear binary codes // Inform. Control. 1972. V. 20. No. 2. P. 182–187.

80. *Khoo K., Gong G., Stinson D. R.* A new family of Gold-like sequences // ISIT — IEEE Int. Symposium on Information Theory (Lausanne, Switzerland, June 30-July 5, 2002). Proc. 2002. P. 181.

81. *Khoo K., Gong G., Stinson D. R.* A new characterization of semi-bent and bent functions on finite fields // Designs, Codes and Cryptography. 2006. V. 38. No. 2. P. 279–295.

82. *Krotov D. S., Avgustinovich S. V.* On the Number of 1-Perfect Binary Codes: A Lower Bound // IEEE Trans. Inform. Theory. 2008. V. 54. No. 4. P. 1760–1765.

83. *Kumar P. V., Scholtz R. A., Welch L. R.* Generalized bent functions and their properties // J. Combin. Theory. Ser. A. 1985. V. 40. No. 1. P. 90–107.

84. http://langevin.univ-tln.fr/project/quartics/ — Classification of Boolean Quartics Forms in eight Variables (Langevin P.). 2008.

85. *Langevin P., Leander G.* Monomial bent functions and Stickelberger's theorem // Finite Fields and Applications. 2008. V. 14. P. 727–742.

86. *Langevin P., Rabizzoni P., Véron P., Zanotti J.-P.* On the number of bent functions with 8 variables // Second International Conference BFCA — Boolean Functions: Cryptography and Applications (Rouen, France, March 13-15, 2006). Proc. 2006. P. 125–135.

87. *Leander N. G.* Monomial bent functions // IEEE Trans. Inform. Theory. 2006. V. 52. No. 2. P. 738–743.

88. *Leander N. G., Langevin P.* On exponents with highly divisible Fourier coefficients and conjectures of Niho and Dobbertin // Algebraic Geometry and its applications (France, May 7–11, 2007) Proc. 2008. P. 410–418.

89. *Maitra S., Sarkar P.* Maximum Nonlinearity of Symmetric Boolean Functions on Odd Number of Variables // IEEE Trans. Inform. Theory. 2002. V. 48. No. 9. P. 2626–2630.

90. *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 (Lecture Notes in Comput. Sci. V. 765).

91. *McFarland R. L.* A family of difference sets in non-cyclic groups // J. Combin. Theory. Ser. A. 1973. V. 15. No. 1. P. 1–10.

92. *Meng Q., Yang M. C., Zhang H., Cui J.-S.* A novel algorithm enumerating bent functions // Cryptology ePrint Archive, Report 2004/274, available at http://eprint.iacr.org/.

93. *Meng Q., Zhang H., Wang Z.* Designing bent functions using evolving computing // Acta Electronica Sinica. 2004. No. 11. P. 1901–1903.

94. *Millan W., Clark A., Dawson E.* An effective genetic algorithm for finding highly nonlinear Boolean functions // First Int. conference on Information and Communications Security — ICICS'97. (Beijing, China, November 11–14, 1997). Proc. Springer Verlag, 1997. P. 149–158 (Lecture Notes in Comput. Sci. V. 1334).

95. *Millan W., Clark A., Dawson E.* Smart hill climbing finds better Boolean functions // Workshop on Selected Areas in Cryptology. 1997. Workshop record. P. 50–63.

96. *Nyberg K.* Perfect nonlinear S-boxes // Advances in cryptology — EUROCRYPT'1991. Int. conference on the theory and application of cryptographic techniques (Brighton, UK, April 8–11, 1991). Proc. Berlin: Springer, 1991. P. 378–386 (Lecture Notes in Comput. Sci. V. 547).

97. *Nyberg K.* Differentially uniform mappings for cryptography // Advances in cryptology — EUROCRYPT'1993. Int. conference on the theory and application of cryptographic techniques (Lofthus, Norway. May 23–27, 1993). Proc. Berlin: Springer, 1994. P. 55–64 (Lecture Notes in Comput. Sci. V. 765).

98. *Olejár D., Stanek M.* On cryptographic properties of random Boolean functions // J. Universal Computer Science. 1998. V. 4. No. 8. P. 705–717.

99. *Olsen J. D., Scholtz R. A., Welch L. R.* Bent-function sequences // IEEE Trans. Inform. Theory. 1982. V. 28. No. 6. P. 858–864.

100. *Parker M. G.* The constabent properties of Golay-Davis-Jedwab sequences // IEEE International Symposium on Information Theory — ISIT'2000. (Sorrento, Italy, June 25-30, 2000). Proc. 2000. P. 302.

101. *Parker M. G., Pott A.* On Boolean functions which are bent and negabent // Sequences, Subsequences, and Consequences — SSC 2007 — International Workshop. (Los Angeles,

CA, USA, May 31 – June 2, 2007). Proc. Berlin: Springer, 2007. P. 9–23 (Lecture Notes in Comput. Sci. V. 4893).

102. *Paterson K. G.* Sequences For OFDM and Multi-code CDMA: two problems in algebraic Coding Theory // Sequences and their applications. – Seta 2001. Second Int. Conference (Bergen, Norway, May 13–17, 2001). Proc. Berlin: Springer, 2002. P. 46–71.
103. *Preneel B., Van Leekwijck W., Van Linden L., Govaerts R., Vandevalle J.* Propogation characteristics of Boolean functions // Advances in cryptology — EUROCRYPT'1990. Int. conference on the theory and application of cryptographic techniques (Aarhus, Denmark, May 21–24, 1990). Proc. Berlin: Springer, 1991. P. 161–173 (Lecture Notes in Comput. Sci. V. 473).
104. *Preneel B.* Analysis and design of cryptographic hash functions // Ph. D. thesis, Katholieke Universiteit Leuven, 3001 Leuven, Belgium. 1993.
105. *Qu C., Seberry J., Pieprzyk J.* Homogeneous bent functions // Discrete Appl. Math. 2000. V. 102. No. 1-2. P. 133–139.
106. *Riera C., Parker M. G.* Generalised Bent Criteria for Boolean Functions (I) // IEEE Trans. Inform. Theory 2006. V. 52. No. 9. P. 4142–4159.
107. *Rodier F.* Asymptotic nonlinearity of Boolean functions // Designs, Codes and Cryptography. 2006. V. 40. No. 1. P. 59–70. Preprint is available at `http://iml.univ-mrs.fr/editions/preprint2003/files/RodierFoncBool.pdf`
108. *Rodier F.* Private Communication. 2008.
109. *Rothaus O.* On bent functions // IDA CRD W.P. No. 169. 1966.
110. *Rothaus O.* On bent functions // J. Combin. Theory. Ser. A. 1976. V. 20. No. 3. P. 300–305.
111. *Schmidt K-U.* Quaternary Constant-Amplitude Codes for Multicode CDMA // Available at `http://arxiv.org/abs/cs.IT/0611162`.
112. *Wang L., Zhang J.* A best possible computable upper bound on bent functions // J. West of China. 2004. V. 33. No. 2. P. 113–115.
113. *Yang M., Meng Q., Zhang H.* Evolutionary design of trace form bent functions // Cryptology ePrint Archive, Report 2005/322, available at `http://eprint.iacr.org/`.
114. *Youssef A., Gong G.* Hyper-bent functions // Advances in cryptology — EUROCRYPT'2001. Int. conference on the theory and application of cryptographic techniques (Innsbruk, Austria, May 6–10, 2001). Proc. Berlin: Springer, 2001. P. 406–419 (Lecture Notes in Comput. Sci. V. 2045).
115. *Yu N. Y., Gong G.* Constructions of quadratic bent functions in polynomial forms // IEEE Trans. Inform. Theory 2006. V. 52. No. 7. P. 3291–3299.

2009 Теоретические основы прикладной дискретной математики №1(3)
DOI 10.17223/20710410/3/3
УДК 519.816

ЛОГИЧЕСКИЕ МЕТОДЫ ПОСТРОЕНИЯ И АНАЛИЗА МОДЕЛЕЙ ВЫБОРА[1]

Л. А. Шоломов

Институт системного анализа РАН, г. Москва

E-mail: sholomov@isa.ru

Логические методы используют представление функций выбора и моделей выбора в виде формул некоторого логического языка и сводят задачи построения и исследования моделей выбора к формальным преобразованиям и анализу этих представлений. Они позволяют решать широкий круг конструктивных задач, связанных с построением, анализом, упрощением и оценкой сложности формальных моделей выбора, дают возможность применить к задачам выбора методологию Шеннона—Яблонского—Лупанова изучения управляющих систем. В данной статье систематизируются результаты автора по исследованию моделей выбора логическими методами, полученные в разное время и опубликованные в разных изданиях. Приводятся относящиеся к этой области результаты других авторов.

Ключевые слова: *функция выбора, модель выбора, анализ, синтез, аппроксимация, оптимизация, оценка сложности, последовательный выбор, параллельный выбор, оператор группового выбора, декомпозиция, агрегирование, порядковое отношение, порядковая модель.*

Введение

В связи с широким применением вычислительной техники в процедурах принятия и реализации решений повышается роль формальных методов исследования моделей выбора вариантов. При построении и исследовании моделей выбора возникает ряд задач конструктивного характера. Основной является задача синтеза — построения модели из заданного класса, реализующей требуемый выбор. При синтезе существенная роль отводится сложности моделей и обычно решается задача построения моделей малой (лучше минимальной) сложности. Иногда ставят задачу минимизации — нахождения по заданной модели эквивалентной ей модели наименьшей сложности. Если выбор не представим моделями рассматриваемого класса, возникает задача его аппроксимации в этом классе (наилучшей в условленном смысле). Важную роль играет задача анализа моделей — выяснения, обладает ли модель или реализуемый ей выбор требуемыми свойствами. Кроме этого рассматриваются задачи агрегирования нескольких моделей выбора в одну и декомпозиции модели на более простые.

В теории и приложениях имеют дело с двумя основными типами моделей выбора. В моделях первого типа варианты считаются некоторыми целостными объектами, не имеющими структурного описания, и предпочтительность одного варианта другому выявляется на основе их непосредственного сравнения. Выбор в рамках таких моделей будем называть свободным в соответствии с термином «свободный», принятым в математике и означающим отсутствие дополнительных соотношений. В моделях второго типа вариантам сопоставляются оценки по заданной совокупности критериев, и

[1]Работа выполнена при финансовой поддержке ОНИТ РАН по программе «Фундаментальные основы информационных технологий и систем».

варианты сравниваются на основе этих оценок. Выбор такого типа называют многокритериальным. В качестве моделей многокритериального выбора часто используются порядковые модели, в которых важны не сами величины оценок по критериям, а их соотношения (больше, меньше, равно).

Логические методы используют представление функций и моделей выбора в виде формул некоторого логического языка и сводят задачи, связанные с построением и анализом моделей, к формальным преобразованиям и исследованию этих представлений. Вид применяемого логического языка зависит от типа моделей. В задачах свободного выбора с конечным числом вариантов в качестве логического языка используется язык булевой алгебры. В случае, когда число вариантов потенциально бесконечно, логические описания требуют привлечения кванторов и применяется некоторый вариант языка первого порядка. В частности, это относится к многокритериальному (в том числе порядковому) выбору, где множество вариантов (наборов значений критериев), вообще говоря, бесконечно. Поскольку результат сравнения оценок может иметь три исхода (больше, меньше, равно), исследование свойств порядковых отношений основывается на языке (3,2)-значной логики.

Логические методы оказались весьма эффективными в силу ряда причин. Описание многих моделй выбора использует теоретико-множественные операции над множествами вариантов, различные логические связки и кванторы. Поскольку имеется естественное соответствие между теоретико-множественными и логическими операциями, а связки и кванторы присутствуют в логическом языке, такие описания легко переводятся в формулы языка. Часто более сложные модели выбора строятся из простых с помощью некоторых операций (агрегирование отношений, параллельное и последовательное соединение моделей и др.). Этим операциям соответствуют определенные преобразования логических формул, что дает возможность находить логические представления достаточно сложных иерархических моделей выбора. Богатая система формальных преобразований позволяет переходить от одних записей в логическом языке к другим, существенно упрощать представления. Имеется возможность привлечения результатов, методов и технических приемов, развитых в математической логике и теории моделей. Логические методы позволяют применять к задачам выбора методологию исследования управляющих систем, заложенную в работах К. Э. Шеннона и получившую существенное развитие в научной школе С. В. Яблонского и О. Б. Лупанова. Все это делает логические методы удобным и универсальным инструментом для решения широкого круга конструктивных задач в области выбора. Впервые логическое описание функций (но не моделей) выбора использовалось в [1]. Логические методы исследования моделей выбора ведут начало от [2]. Различные варианты логических методов для задач выбора применялись в [3–6].

Важную роль при решении конструктивных задач в области выбора играют сложностные аспекты, и им уделяется значительное внимание в проводимых исследованиях. Рассматриваются два типа характеристик сложности:

- сложность моделей выбора, измеряемаяемая каким-либо существенным параметром, связанным с описанием или структурой моделей;

- сложность (трудность) задач, относящихся к построению либо анализу моделей выбора, характеризуемая размером какого-либо вычислительного ресурса (времени, памяти), затрачиваемого на их решение.

В области оценок сложности моделей выбора большое внимание уделено асимптотическим результатам. Они дают возможность получить качественное представление

о поведении сложности, выяснить влияние на нее тех или иных параметров задачи выбора, зависимость от класса используемых моделей, взаимоотношение различных характеристик сложности. При исследовании трудности задач основная роль отведена их анализу на полиномиальность, NP-полноту и алгоритмическую неразрешимость. Согласно существующей точке зрения считается, что задачи с полиномиальной оценкой трудоемкости эффективно решаемы, NP-полные задачи труднорешаемы, алгоритмически неразрешимые задачи в принципе не решаемы.

Часть I. МОДЕЛИ СВОБОДНОГО ВЫБОРА

1. Логическое представление свободного выбора

1.1. Функции выбора

Пусть задано множество A *вариантов* (конечное или бесконечное). Подмножества $X \subseteq A$ называются *предъявлениями* (более точно — множествами, предъявленными для выбора). *Функция выбора* (ФВ) C на множестве вариантов A представляет собой отображение $2^A \to 2^A$, сопоставляющее каждому $X \subseteq A$ множество вариантов $C(X) \subseteq X$, *выбранных* в предъявлении X.

Более общим понятием является *функция неполного выбора* (ФНВ) [2, 7], которая задается множеством $\mathcal{A} \subseteq 2^A$ *допустимых предъявлений* и функциями $C^1, C^0 : \mathcal{A} \to 2^A$ такими, что $C^1(X) \subseteq X$, $C^0(X) \subseteq X$, $C^1(X) \cap C^0(X) = \varnothing$ для любого $X \in \mathcal{A}$. Варианты из множества $C^1(X)$ называются *принятыми* (в предъявлении X), из множества $C^0(X)$ — *отвергнутыми*. Такую ФНВ будем обозначать $\langle \mathcal{A}, C^1, C^0 \rangle$. Поскольку всякую ФВ C можно рассматривать как ФНВ при $\mathcal{A} = 2^A$, $C^1(X) = C(X)$ и $C^0(X) = X \setminus C(X)$, все дальнейшие понятия будут формулироваться применительно к ФНВ.

Доопределением ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$ будем называть любую ФВ C (всюду определенную), для которой $C^1(X) \subseteq C(X) \subseteq X \setminus C^0(X)$, $X \in \mathcal{A}$. Всякой модели (механизму, правилу, процедуре) выбора M соответствует некоторая ФВ C_M, которая сопоставляет каждому предъявлению $X \subseteq A$ множество вариантов $C_M(X)$, выбранных из X моделью M. Скажем, что модель M *реализует* ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$, если C_M является доопределением этой функции.

1.2. Логическое представление функций выбора

Пока не оговорено противное, будем рассматривать модели свободного выбора с конечным множеством вариантов A; пусть его мощность равна n. Не различая варианты и их номера, будем полагать $A = A^{(n)} = \{1, \ldots, n\}$. С каждым вариантом i свяжем булеву переменную x_i, и предъявление X будем задавать набором $\tilde{X} = (x_1, \ldots, x_n)$, где $x_i = 1$ и $x_i = 0$ соответственно при $i \in X$ и $i \notin X$. ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$ будем описывать n частичными булевыми функциями $C^{(i)}(\tilde{X})$, $1 \leqslant i \leqslant n$ [2, 7], где

$$C^{(i)}(\tilde{X}) = \begin{cases} 1, & \text{если } X \in \mathcal{A} \wedge i \in C^1(X), \\ 0, & \text{если } (X \in \mathcal{A} \wedge i \in C^0(X)) \vee i \notin X, \\ * & \text{в остальных случаях.} \end{cases}$$

Здесь $*$ обозначает неопределенное значение. Совокупность n функций $C^{(i)}(\tilde{X})$ будем называть *логическим представлением* функции неполного выбора. Включение $C^1(X) \subseteq X$ эквивалентно тому, что каждая функция $C^{(i)}(\tilde{X})$ выразима в виде $C^{(i)}(\tilde{X}) = x_i C^{(i)}(\tilde{X}^{(i)})$, где $\tilde{X}^{(i)} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$, $C^{(i)}(\tilde{X}^{(i)})$ — частичная булева функция, полученная из $C^{(i)}(\tilde{X})$ подстановкой $x_i = 1$. В случае ФВ представляющие булевы функции всюду определены.

Модель выбора M реализует ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$ тогда и только тогда, когда при каждом i булева функция $C_M^{(i)}(\tilde{X})$ доопределяет частичную функцию $C^{(i)}(\tilde{X})$.

1.3. Логическое представление выбора по отношению

Обычно модели выбора строятся с использованием бинарных отношений $r \subseteq A^2$ (на множестве вариантов A), в которых соотношение xry интерпретируется как «вариант x предпочтительнее y». Для $i \in A$ обозначим $r(i) = \{j \mid irj\}$, $r^{-1}(i) = \{j \mid jri\}$. В теории и приложениях используются различные правила выбора по бинарным отношениям (см., например, [8, 9]). Приведем три из них:

- выбор *максимальных* вариантов $C_{r,1}(X) = \{x \in X \mid (\forall y \in X) y \bar{r} x\}$;
- выбор *лучших* вариантов $C_{r,2}(X) = \{x \in X \mid (\forall y \in X) xry\}$;
- выбор *отбраковкой худших* вариантов $C_{r,3}(X) = X \setminus \{x \in X \mid (\forall y \in X) x \bar{r} y \wedge (\exists z \in X) zrx\}$.

В первом и третьем правилах отношение предполагается иррефлексивным ($x \bar{r} x$), во втором — рефлексивным (xrx). Указанные правила имеют следующие логические представления [7]:

$$C_{r,1}^{(i)}(\tilde{X}) = x_i \bigwedge_{j \in r^{-1}(i)} \bar{x}_j, \quad C_{r,2}^{(i)}(\tilde{X}) = x_i \bigwedge_{j \in \bar{r}(i)} \bar{x}_j, \quad C_{r,3}^{(i)}(\tilde{X}) = x_i \Big(\bigvee_{l \in r(i)} x_l \vee \bigvee_{j \in r^{-1}(i)} \bar{x}_j \Big).$$

Двойственным отношением к r называется отношение $r^* = \bar{r}^{-1}$ (операции $^-$ и $^{-1}$ перестановочны и их порядок не существен). Правила выбора максимальных и лучших вариантов сводятся друг к другу путем перехода к двойственному отношению: $C_{r,1} = C_{r^*,2}$, $C_{r,2} = C_{r^*,1}$. Основным правилом выбора по отношению обычно считают первое, и, если не оговорено противное, будем под выбором по отношению понимать $C_{r,1}$ и использовать для него обозначение C_r. Класс всех моделей выбора по отношению (использующих первое правило) обозначим через **Rl**.

1.4. Суперпозиция логических представлений

Данный раздел написан по материалам [2, 10, 11], систематическое изложение имеется в [7].

Введем некоторые операции над функциями и моделями выбора. Операция *суперпозиции* $C_1 \circ C_2$ функций выбора C_1 и C_2 задается равенством $(C_1 \circ C_2)(X) = C_2(C_1(X))$. Логическое представление суперпозиции выражается через логические представления исходных ФВ в виде

$$(C_1 \circ C_2)^{(i)}(\tilde{X}) = C_2^{(i)}(C_1^{(1)}(\tilde{X}), \ldots, C_1^{(n)}(\tilde{X})).$$

Эта операция возникает при последовательном соединении моделей выбора M_1 и M_2, когда выход модели M_1 подается на вход M_2. Полученная модель реализует ФВ $C_{M_1 M_2} = C_{M_1} \circ C_{M_2}$.

Операция суперпозиции ФВ ассоциативна: $(C_1 \circ C_2) \circ C_3 = C_1 \circ (C_2 \circ C_3)$, поэтому можно рассматривать многоместную операцию $C_1 \circ C_2 \circ \cdots \circ C_k$, которая не зависит от расстановки скобок. Последовательное соединение k моделей выбора по отношению дает *модель последовательного выбора глубины* k. Для нее $C_{r_1 r_2 \ldots r_k} = C_{r_1} \circ C_{r_2} \circ \cdots \circ C_{r_k}$, или в другой записи

$$C_{r_1 r_2 \ldots r_k}(X) = C_{r_k}(\ldots C_{r_2}(C_{r_1}(X)) \ldots).$$

С использованием приведенных выше формул можно найти логическое представление этой модели. В частности, для моделей глубины 2 оно преобразуется к виду

$$C_{r_1r_2}(\tilde{X}) = x_i \bigwedge_{j\in r_1^{-1}(i),\, m\in r_2^{-1}(i)} \bar{x}_j\Big(\bar{x}_m \vee \bigvee_{l\in r_1^{-1}(m)} x_l\Big).$$

Классы моделей последовательного выбора произвольной глубины и глубины k обозначим соответственно через $\mathbf{Sq}$ и $\mathbf{Sq}_k$.

1.5. Композиция логических представлений

Операция *композиции* функций выбора $C_1, \ldots, C_k$ при *способе композиции* $F = F(Y_1, \ldots, Y_k)$ реализует ФВ $C(X) = F(C_1(X), \ldots, C_k(X))$. В качестве F обычно используется монотонная теоретико-множественная операция (операция голосования). В этом случае логическое представление композиции имеет вид

$$C^{(i)}(\tilde{X}) = f(C_1^{(i)}(\tilde{X}), \ldots, C_k^{(i)}(\tilde{X})),$$

где $f = f(y_1, \ldots, y_k)$ — монотонная булева функция, естественным образом сопоставленная операции F. Операция композиции возникает при параллельном соединении моделей $M_1, \ldots, M_k$, когда их выходы подаются на вход блока, реализующего k-местный оператор F. Если в качестве $M_1, \ldots, M_k$ использовать модели выбора по отношениям $r_1, \ldots, r_k$, получим *модель параллельного выбора ширины* k. Ей соответствует логическое описание

$$C^{(i)}_{F,r_1,\ldots,r_k}(\tilde{X}) = f(C^{(i)}_{r_1}(\tilde{X}), \ldots, C^{(i)}_{r_k}(\tilde{X})).$$

Класс моделей параллельного выбора обозначим через $\mathbf{Pr}$.

С помощью введенных операций над моделями выбора можно определить более сложные модели — последовательно-параллельного выбора [12], обобщенного математического программирования [13] и др. — и найти их логическое описание.

Помимо суперпозиции и композиции используются и другие операции над моделями выбора. В их числе операция ветвления, когда в зависимости от наличия некоторого свойства применяется та или иная модель выбора, операция ограниченной обратной связи, при которой выбор производится в несколько туров, каждый из которых зависит от результатов предыдущего тура. Для реализации этих операций используются более сильные логические средства — логика первого порядка [14] (см. п. 6.2). Весьма распространенным способом выбора на основе нескольких отношений является выбор по некоторому отношению, образованному их агрегированием. Исследование этой модели производится в рамках языка (3,2)-значной логики, развитого для порядковых отношений (см. п. 8.2).

2. Сложность задач, связанных с анализом и синтезом моделей

2.1. Формулировка задач

Пусть задан некоторый класс $\mathbf{M}$ моделей выбора. Будем говорить, что ФНВ $\langle \mathcal{A}, C^1, C^0\rangle$ *представима в классе* $\mathbf{M}$, если она реализуется некоторой моделью этого класса. *Задача синтеза* в классе $\mathbf{M}$ состоит в том, чтобы по заданной ФНВ установить, представима ли она в этом классе, и, если представима, — построить реализующую ее модель $M \in \mathbf{M}$.

ФВ C^* (ФВ C_*) называется *мажорантой* (*минорантой*) заданной ФНВ $\langle \mathcal{A}, C^1, C^0\rangle$, если для любого $X \in \mathcal{A}$ выполнено $C^1(X) \subseteq C^*(X)$ $(C^0(X) \subseteq X \setminus C_*(X))$. ФВ C^+

(ФВ C^-) называется *верхней* (*нижней*) *аппроксимацией* заданной ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$ в классе моделей $\mathbf{M}$, если C^+ — мажоранта (C^- — миноранта) этой ФНВ, представимая в классе $\mathbf{M}$, и для всякой ее мажоранты C^* (миноранты C_*), представимой в $\mathbf{M}$, при всех X выполнено включение $C^*(X) \supseteq C^+(X)$ ($C_*(X) \subseteq C^-(X)$). Легко видеть, что верхняя (нижняя) аппроксимация, если она существует, — единственна. *Задача об аппроксимации* (верхней или нижней) в классе $\mathbf{M}$ состоит в том, чтобы по ФНВ узнать, имеет ли она соответствующую аппроксимацию в этом классе, и, если имеет, — найти ее.

Пусть с моделью M связана некоторая характеристика *сложности* $l(M)$. *Задача оптимального синтеза* в классе моделей $\mathbf{M}$ ставится как задача построения по ФНВ реализующей ее модели $M \in \mathbf{M}$ с минимальным значением $l(M)$. Модели M_1 и M_2 будем называть *эквивалентными*, если $C_{M_1} = C_{M_2}$. *Задача минимизации* в классе $\mathbf{M}$ состоит в том, чтобы по модели $M \in \mathbf{M}$ построить эквивалентную модель $M' \in \mathbf{M}$, имеющую наименьшую сложность.

Приведем результаты по исследованию перечисленных задач на NP-трудность и полиномиальность применительно к классам моделей **Rl**, **Sq** и **Pr**, введенным выше.

2.2. Результаты для класса **Rl**

Под сложностью $l(r)$ отношения r будем понимать число входящих в него пар. Пусть задана ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$. С каждым i, $1 \leqslant i \leqslant n$, свяжем множество

$$\Sigma_i = \bigcup_{X \mid i \in C^1(X)} X.$$

Если оно пусто, полагаем $\Sigma_i = \{i\}$. Образуем отношение $\check{r}$, задав $\check{r}^{-1}(i) = A \setminus \Sigma_i$, $1 \leqslant i \leqslant n$. Кроме того, построим *парно-выявленное отношение* $\hat{r}$, где $i \hat{r} j \iff \{i, j\} \in \mathcal{A} \wedge j \in C^0(\{i, j\})$.

Теорема 1 [7, 10].

1. ФНВ $\langle \mathcal{A}, C^1, C^0 \rangle$ представима в классе **Rl** тогда и только тогда, когда $(\forall X \in \mathcal{A})(X \subseteq \Sigma_i \Rightarrow i \notin C^0(X))$, и при выполнении этого условия в качестве ее реализации может быть взято отношение $\check{r}$.

2. Всякая ФНВ имеет в классе **Rl** верхнюю аппроксимацию, которая реализуется отношением $\check{r}$.

3. Нижняя аппроксимация в классе **Rl** для ФНВ существует тогда и только тогда, когда $(\forall X \in \mathcal{A})(C_{\hat{r}}(X) \cap C^0(X) = \varnothing)$, и при выполнении этого условия нижняя аппроксимация реализуется отношением $\hat{r}$.

4. Задача оптимального синтеза в классе **Rl** является NP-трудной.

Для всюду определенных ФВ п. 3 теоремы доказан в [15]. Теорема показывает, что задачи синтеза и аппроксимации в классе **Rl** полиномиальны, а оптимального синтеза — NP-трудна. Задача минимизации в классе **Rl** не возникает, поскольку разным отношениям соответствуют разные ФВ. Отметим, что если рассматривать выбор лучших вариантов (правило 2), то все перечисленные задачи (включая задачу оптимального синтеза) оказываются полиномиальными [7].

2.3. Результаты для класса **Sq**

Модель M последовательного выбора задается набором отношений $(r_1, \ldots, r_k)$. С такой моделью M будем связывать две сложностные характеристики: $l(M)$ — суммарное число пар в отношениях модели M и $k(M)$ — число отношений в M. Эти характеристики будем называть соответственно *сложностью* и *глубиной* модели. Для

уменьшения сложности модели можно использовать операцию *приведения*, при выполнении которой для каждых $i, j \in A$ связывающие их пары (i, j) и (j, i) удаляются из всех отношений, кроме того, где они встретились впервые. Приведенная модель эквивалентна исходной [10].

При решении конструктивных задач для моделей последовательного выбора возникают трудности принципиального характера, поскольку в большинстве своем эти задачи оказываются NP-трудными.

Теорема 2 [7, 10, 16].

1. При любом $k \geqslant 2$ задача синтеза моделей в классе $\mathbf{Sq}_k$ NP-трудна.

2. При любом $k \geqslant 2$ задачи нахождения верхней и нижней аппроксимаций в классе $\mathbf{Sq}_k$ NP-трудны.

3. При любом k задача минимизации сложности в классе $\mathbf{Sq}_k$ полиномиальна: минимальной по сложности является приведенная модель.

4. При любом $k \geqslant 3$ задача минимизации глубины в классе $\mathbf{Sq}_k$ NP-трудна. При $k = 2$ она полиномиальна.

NP-трудность задач синтеза и аппроксимации в классе $\mathbf{Sq}_k$ объясняется тем, что задача о реализуемости ФНВ в этом классе является NP-трудной (данный факт доказан в [7] для $k = 2$, но может быть распространен на произвольные k). В связи с выявленной трудностью этих задач рассматривались некоторые их ослабленные постановки, для которых удалось найти эффективные способы решения. В [7] описан метод последовательной аппроксимации, который с полиномиальной трудоемкостью при определенных статистических предположениях относительно множества допустимых предъявлений $\mathcal{A}$ строит при каждом k для почти всех (при $n \to \infty$) функций, представимых в $\mathbf{Sq}_k$, модели, одновременно лучшие по сложности и глубине. В [16] изучен некоторый ослабленный вариант задачи минимизации глубины (задача сжатия), и в случае, когда модель использует лишь ацикличные отношения (определение см. в п. 4.2), предложен полиномиальный алгоритм ее решения. Отметим, что в прикладных задачах обычно применяются ацикличные отношения, ибо лишь они гарантируют непустоту выбора.

2.4. Результаты для класса **Pr**

Пусть модель M параллельного выбора использует отношения $r_1, \ldots, r_k$ и оператор F. С такой моделью M будем связывать две сложностные характеристики: $l(M)$ — суммарное число пар в отношениях модели M и $k(M)$ — число отношений в M. Эти характеристики будем называть соответственно *сложностью* и *шириной* модели.

В большинстве своем конструктивные задачи, относящиеся к модели **Pr**, оказываются эффективно решаемыми. В определенной степени это объясняется тем, что имеется простой алгоритм проверки представимости ФНВ в классе **Pr**, основанный на том, что необходимым и достаточным условием представимости в классе **Pr** является антимонотонность частичных булевых функций $C^{(i)}(\tilde{X}^{(i)})$, $1 \leqslant i \leqslant n$ (антимонотонные функции — отрицания монотонных) [7, 11].

Теорема 3 [7, 11].

1. Задача синтеза моделей в классе **Pr** полиномиальна.

2. Верхняя и нижняя аппроксимации в классе **Pr** существуют для всех ФНВ, и задача их построения полиномиальна.

3. Задача минимизации сложности в классе **Pr** полиномиальна.

4. Задача минимального синтеза в классе **Pr** NP-трудна.

Нахождение верхней (нижней) аппроксимации в классе **Pr** сводится к построению минимальных (максимальных) всюду определенных антимонотонных функций, не меньших (не больших) $C^{(i)}(\tilde{X}^{(i)})$. В случае верхней аппроксимации такие функции имеют вид

$$\check{C}^{(i)}(\tilde{X}^{(i)}) = \bigvee_{Z|\ Z=\{i\}\vee Z\in\mathcal{A}\wedge i\in C^1(Z)} \bigwedge_{j\in A\setminus Z} \bar{x}_j, \qquad 1\leqslant i\leqslant n.$$

Для нижней аппроксимации функции строятся двойственным образом. Если ФНВ представима в классе **Pr** (т. е. функции $C^{(i)}(\tilde{X}^{(i)})$ антимонотонны), то в качестве ее реализации может быть взята модель, реализующая верхнюю аппроксимацию.

Что касается задачи минимизации ширины модели (которая в теореме не упомянута), то скорее всего она NP-трудна, ибо к ней полиномиально сводится задача о минимальном дизъюнктивном базисе системы множеств, не сравнимых по включению [7]. Задача о минимальном дизъюнктивном базисе NP-трудна [17] и, по-видимому, то же справедливо при условии несравнимости множеств системы.

3. Сложность моделей выбора

3.1. Характеристики сложности

Пусть $\mathbf{M}$ — некоторый класс моделей выбора и $\mathfrak{C}_{\mathbf{M}}^{(n)}$ — класс ФВ на n-элементном множестве вариантов $A^{(n)}$, представимых моделями из $\mathbf{M}$. Пусть с каждой моделью $M\in\mathbf{M}$ связана некоторая характеристика сложности $l(M)$. *Сложностью функции выбора* $C\in\mathfrak{C}_{\mathbf{M}}^{(n)}$ (в классе моделей $\mathbf{M}$) называется величина

$$l_{\mathbf{M}}(C) = \min\{l(M)\mid M\in\mathbf{M},\ C_M = C\}.$$

Сложность класса моделей $\mathbf{M}$ характеризуется функцией

$$l_{\mathbf{M}}(n) = \max\{l_{\mathbf{M}}(C)\mid C\in\mathfrak{C}_{\mathbf{M}}^{(n)}\}.$$

Возникает задача исследования поведения функции $l_{\mathbf{M}}(n)$ и разработки методов синтеза, гарантирующих оценки, близкие к $l_{\mathbf{M}}(n)$. Ставятся и другие задачи оценочного характера. Одна из них связана с изучением взаимного влияния различных характеристик сложности. Пусть помимо $l(M)$ задана некоторая характеристика сложности $k(M)$ моделей $M\in\mathbf{M}$. Введем величину

$$l^*_{\mathbf{M}}(C) = \min\{l(M)\mid M\in\mathbf{M},\ C_M = C,\ k(M) = k_{\mathbf{M}}(C)\}.$$

Сравнение величин $l^*_{\mathbf{M}}(C)$ с $l_{\mathbf{M}}(C)$ показывает, к какому увеличению сложности l приводит минимизация k. Аналогично может быть введена величина $k^*_{\mathbf{M}}(C)$ и на ее основе изучено влияние l на k.

Исследуем поведение сложностных характеристик для классов моделей **Sq** и **Pr**. Как и раньше, для моделей M из этих классов в качестве характеристик сложности будем рассматривать суммарное число $l(M)$ пар в отношениях модели M и число отношений $k(M)$. Отметим, что в классе **Rl** нетривиальной характеристикой является лишь $l_{\mathbf{Rl}}(n)$, ибо $k_{\mathbf{Rl}}(n) = 1$. Из достаточно простых соображений следует, что $l_{\mathbf{Rl}}(n) = n(n-1)$.

3.2. Сложность моделей класса **Sq**

Асимптотически точные значения сложности и глубины моделей класса **Sq** даются следующим утверждением.

Теорема 4 [7, 10].

$$l_{\mathbf{Sq}}(n) \sim n^2, \qquad k_{\mathbf{Sq}}(n) \sim n^2/2.$$

Следует отметить, что обе асимптотики могут быть достигнуты одновременно (в одной модели).

Для выяснения взаимного влияния параметров реализации в классе **Sq** рассмотрим величины $l^*_{\mathbf{Sq}}(C)$ и $k^*_{\mathbf{Sq}}(C)$. Если обозначить через $l^*_{\mathbf{Sq}}(n)$ и $k^*_{\mathbf{Sq}}(n)$ их максимумы по всем $C \in \mathfrak{C}^{(n)}_{\mathbf{Sq}}$, то можно показать, что они совпадают с $l_{\mathbf{Sq}}(n)$ и $k_{\mathbf{Sq}}(n)$. Поэтому для изучения взаимного влияния $l(M)$ и $k(M)$ в классе **Sq** будем использовать другие функции. Введем величины

$$\lambda^l_{\mathbf{Sq}}(C) = l^*_{\mathbf{Sq}}(C)/l_{\mathbf{Sq}}(C), \qquad \lambda^l_{\mathbf{Sq}}(n) = \max\{\lambda^l_{\mathbf{Sq}}(C) \mid C \in \mathfrak{C}^{(n)}_{\mathbf{Sq}}\},$$

характеризующие увеличение сложности при минимизации глубины. Аналогично введем характеристики $\lambda^k_{\mathbf{Sq}}(C)$ и $\lambda^k_{\mathbf{Sq}}(n)$ увеличения глубины при минимизации сложности.

Теорема 5 [7, 10].

$$\lambda^l_{\mathbf{Sq}}(n) \sim n, \qquad n^2/4 \lesssim \lambda^k_{\mathbf{Sq}}(n) \leqslant n^2/2.$$

Доказательство теоремы показывает, что асимптотически максимальное увеличение сложности может произойти при уменьшении глубины всего на 1, а максимальное по порядку увеличение глубины — при небольшом уменьшении сложности.

3.3. Сложность моделей класса **Pr**

Асимптотически точные значения сложности и ширины моделей класса **Pr** приводятся в следующем утверждении.

Теорема 6 [7, 11].

$$l_{\mathbf{Pr}}(n) \sim n^2, \qquad k_{\mathbf{Pr}}(n) \sim n.$$

В отличие от класса **Sq** здесь не удалось совместить обе асимптотики в одной конструкции. Однако для произвольной функции C из $\mathfrak{C}^{(n)}_{\mathbf{Pr}}$ построена модель M, параметры которой отличаются от $l_{\mathbf{Sq}}(n)$ и $k_{\mathbf{Sq}}(n)$ не более чем в 2 раза. Более точно, $l(M) \leqslant 2n(n-1)$, $k(M) \leqslant 2n+1$. Кроме того, в случае, когда в модели используются лишь частичные порядки и сложность отношения измеряется числом дуг в базисном графе, в [18] предложена конструкция, асимптотически наилучшая одновременно по сложности и ширине.

Для выяснения взаимного влияния параметров реализации в классе **Pr** рассмотрим величины $l^*_{\mathbf{Pr}}(C)$ и $k^*_{\mathbf{Pr}}(C)$ и обозначим через $l^*_{\mathbf{Pr}}(n)$ и $k^*_{\mathbf{Pr}}(n)$ их максимумы по всем $C \in \mathfrak{C}^{(n)}_{\mathbf{Pr}}$. Получены следующие результаты.

Теорема 7 [7, 11].

$$l^*_{\mathbf{Pr}}(n) \sim n^3, \qquad 0,94n^{3/2} \lesssim k^*_{\mathbf{Pr}}(n).$$

Из них вытекает, что минимизация ширины приводит к увеличению максимальной сложности асимптотически в n раз, а минимизация сложности — к увеличению максимальной ширины моделей по порядку не менее чем в $\sqrt{n}$ раз (легко доказать, что это возрастание не может превзойти n). В конструкции из теоремы асимптотически максимальное увеличение сложности происходит при уменьшении ширины на 1.

Следующая теорема показывает, что для конкретных ФВ минимизация сложности может сопровождаться экспоненциальным увеличением ширины, и такой рост максимален.

Теорема 8 [7]. Для любой функции $C \in \mathfrak{C}^{(n)}_{\mathbf{Pr}}$ выполнено $\log_2 k^*_{\mathbf{Pr}}(C) \leqslant k_{\mathbf{Pr}}(C)$ и существует последовательность функций $C_n \in \mathfrak{C}^{(n)}_{\mathbf{Pr}}$ такая, что $\log_2 k^*_{\mathbf{Pr}}(C_n) \sim k_{\mathbf{Pr}}(C_n)$.

3.4. Полные модели

Пусть $\mathbf{M}$ — некоторая модель выбора. Обозначим через $H_n(\mathbf{M})$ *энтропию* (двоичный логарифм числа ФВ) класса $\mathfrak{C}^{(n)}_{\mathbf{M}}$, а через H_n — энтропию класса всех ФВ на множестве $A^{(n)}$. Справедливы следующие оценки энтропии [7]: $H_n = n2^{n-1}$, $H_n(\mathbf{Rl}) = n(n-1)$, $H_n(\mathbf{Sq}_k) \sim (n^2 \log_2(3k+1))/2$, $(n^2 \log_2 n)/4 < H_n(\mathbf{Sq}) \lesssim n^2 \log_2 n$, $H_n(\mathbf{Pr}) \sim \sqrt{n/2\pi}2^n$. Из них следует, что в рассматривавшихся выше классах моделей реализуема лишь малая доля ФВ. В связи с этим представляют интерес классы моделей, в которых представимы произвольные ФВ. Такие классы будем называть *полными*.

Примером полного класса может служить класс $\mathbf{Pr}'$ моделей *параллельного выбора на основе отбраковки худших вариантов* [19]. Они отличаются от рассматривавшихся ранее моделей параллельного выбора тем, что выбор по отношениям осуществляется в соответствии с правилом 3 на с. 41. Сложность модели $M \in \mathbf{Pr}'$ будем характеризовать *шириной* — числом $k(M)$ используемых в ней отношений. При естественной интерпретации величина $k(M)$ совпадает с числом участников процедуры выбора.

Теорема 9 [7, 19].

1. Класс моделей $\mathbf{Pr}'$ полон.
2. Справедливы оценки

$$c_1 n \leqslant k_{\mathbf{Pr}'}(n) \leqslant c_2 n,$$

где c_1 и c_2 — некоторые положительные константы.

Из доказательства теоремы следует, что класс $\mathbf{Pr}'$ остается полным, если в нем ограничиться отношениями, в которых варианты делятся на три класса («хорошие», «удовлетворительные», «плохие») и выбор по отношению производится удалением из предъявления группы «плохих» вариантов. Для такого класса остаются справедливыми оценки теоремы.

Изучены и другие модели со свойством полноты (последовательно-параллельный выбор [7, 12], многошаговые схемы обобщенного математического порграммирования [7, 20]), получены оценки их сложностных параметров.

4. Агрегирование отношений

4.1. Задача синтеза агрегирующих операторов

При необходимости выбора по набору отношений $(r_1, \ldots, r_n)$ часто используют модель, в которой некоторым способом строится агрегированное отношение $r = F(r_1, \ldots, r_n)$ и осуществляется выбор по r. Обычно это бывает в задачах группового выбора, поэтому отношения $r_1, \ldots, r_n$ называют *индивидуальными*, а r — *групповым*.

Систематическое изучение операторов группового выбора F ведет начало от исследований К. Эрроу [21, 22], который сформулировал ряд аксиоматических требований к операторам, каждое из которых представляется необходимым, и доказал их несовместность (теорема невозможности Эрроу [23]). Данный факт получил в литературе название «парадокса Эрроу». Критический анализ использованных аксиом привел к рассмотрению других требований, видоизменивших парадокс, но не устранивших его. Позднее изучение парадокса Эрроу приняло более конструктивный характер. Появился ряд исследований, в которых рассматривались совместные наборы условий и был

найден явный вид удовлетворяющих им операторов. Эта задача получила название задачи синтеза операторов группового выбора. Остановимся на ней подробнее.

Задача синтеза операторов группового выбора состоит в том, чтобы найти явный вид операторов F, удовлетворяющих заданным *характеристическим условиям* и *структурным ограничениям* [23]. Характеристические условия представляют собой аксиоматически заданные требования к операторам, структурные ограничения задаются указанием классов отношений $\mathcal{R}_1$ и $\mathcal{R}_2$ (обычно $\mathcal{R}_1 \subseteq \mathcal{R}_2$) и требованием, чтобы при использовании индивидуальных отношений из $\mathcal{R}_1$ групповое принадлежало классу $\mathcal{R}_2$. Будем говорить, что соответствующий оператор имеет тип $\mathcal{R}_1^n \to \mathcal{R}_2$.

Будем рассматривать операторы, удовлетворяющие характеристическим условиям бинарности, нейтральности (к вариантам) и ненавязанности [23]. Такие операторы (и только они) представимы в виде

$$F(r_1, \ldots, r_n) = \Phi(r_1, \ldots, r_n, r_1^{-1}, \ldots, r_n^{-1}),$$

где Φ — нетривиальная (не дающая тождественно пустого или полного множества) операция. Вместо r_i^{-1} удобно иметь дело с двойственными отношениями r_i^*. Заменив r_i^{-1} на $\overline{r_i^*}$ и модифицировав соответствующим образом оператор Φ, будем рассматривать представления

$$F(r_1, \ldots, r_n) = \Phi(r_1, \ldots, r_n, r_1^*, \ldots, r_n^*).$$

Если на оператор F наложено дополнительное условие монотонности [23], то операция Φ монотонна, т. е. выразима через $\cap$ и $\cup$.

Для того чтобы указать структурные ограничения, опишем основные классы отношений, используемых в моделях выбора.

4.2. Классы отношений

Отношение r называется (а) *рефлексивным*, (б) *иррефлексивным*, (в) *асимметричным*, (г) *антисимметричным*, (д) *полным*, (е) *связным*, (ж) *транзитивным*, (з) *негатранзитивным*, (и) *ациклическим*, если оно удовлетворяет условию (а) xrx, (б) $x\bar{r}x$, (в) $xry \Rightarrow y\bar{r}x$, (г) $x \neq y \wedge xry \Rightarrow y\bar{r}x$, (д) $xry \vee yrx$, (е) $x \neq y \Rightarrow xry \vee yrx$, (ж) $xry \wedge yrz \Rightarrow xrz$, (з) $x\bar{r}y \wedge y\bar{r}z \Rightarrow x\bar{r}z$, (и) $x_1rx_2 \wedge x_2rx_3 \wedge \ldots \wedge x_{k-1}rx_k \Rightarrow x_k\bar{r}x_1$, $k = 1, 2, \ldots$

Транзитивное иррефлексивное отношение называется *частичным порядком* (строгим), связный частичный порядок — *линейным порядком*, а негатранзитивное ациклическое отношение — *слабым порядком*. Частичный порядок r со свойством $xry \wedge zrv \Rightarrow xrv \vee zry$ называется *интервальным порядком*, а интервальный порядок со свойством $xry \wedge yrz \Rightarrow xrv \vee vrz$ — *полупорядком*. Указанные типы порядков наиболее распространены в задачах выбора. Это связано с тем, что линейные порядки представимы строгим критерием, слабые порядки — нестрогим критерием, интервальные порядки — критерием с погрешностью, полупорядки — критерием с постоянной погрешностью, частичные порядки — совокупностью критериев [24].

Обозначим соответственно через $\mathcal{P}$, $\mathcal{L}$, $\mathcal{W}$, $\mathcal{I}$, $\mathcal{S}$, $\mathcal{T}$ и $\mathcal{A}$ классы отношений частичного, линейного, слабого, интервального порядка, полупорядка, ациклических и транзитивных отношений. Справедливы строгие включения

$$\mathcal{L} \subset \mathcal{W} \subset \mathcal{S} \subset \mathcal{I} \subset \mathcal{P} \subset (\mathcal{T}, \mathcal{A}).$$

Классы $\mathcal{T}$ и $\mathcal{A}$ не сравнимы по включению и оба содержат $\mathcal{P}$.

В качестве классов $\mathcal{R}_1, \mathcal{R}_2$, задающих структурные ограничения, будем использовать произвольные пары классов из $\{\mathcal{L}, \mathcal{W}, \mathcal{S}, \mathcal{I}, \mathcal{P}, \mathcal{T}, \mathcal{A}\}$, удовлетворяющие включению $\mathcal{R}_1 \subseteq \mathcal{R}_2$. Всего имеется 27 типов таких операторов $\mathcal{R}_1^n \to \mathcal{R}_2$.

Логические средства, используемые для исследования операторов, подобны средствам, применяемым для изучения порядковых отношений, и будут представлены в п. 8.2.

4.3. Явный вид агрегирующих операторов

Чтобы описать явный вид операторов агрегирования, введем некоторые специальные операторы.

Первый из них хорошо известен. Это *оператор лексикографии*

$$\Lambda(r_1, \ldots, r_n) = \bigcup_{j=1}^{k} \Big(r_j \cap \bigcap_{i=1}^{j-1} r_i^* \Big), \qquad k \in \{1, \ldots, n\}.$$

Оператор транзитивной лексикографии является оператором одного из двух типов [25]. Первый имеет вид

$$T(r_1, \ldots, r_n) = \bigcup_{j=1}^{k} \Big(r_j^* \cap \bigcap_{i=1}^{j-1} r_i \Big), \qquad k \in \{1, \ldots, n\},$$

второй получается из него заменой члена $r_1 \cap \ldots \cap r_{k-1} \cap r_k^*$ на $r_1 \cap \ldots \cap r_k$ (такой оператор также будем обозначать T). Дальше, говоря об операторах Λ и T, будем понимать операторы указанного вида, в которых вместо $r_1, \ldots, r_k$ могут быть использованы произвольные отношения $r_{i_1}, \ldots, r_{i_k}$ набора $(r_1, \ldots, r_n)$.

Чтобы описать следующий тип операторов [26], дадим ряд определений. Булева функция $f(x_1, \ldots, x_n)$ называется *пороговой*, если существуют такие числа $w_1, \ldots, w_n$ (*веса*) и t (*порог*), что

$$f(x_1, \ldots, x_n) = 1 \Leftrightarrow w_1 x_1 + \ldots + w_n \geqslant 1.$$

Набор $(w_1, \ldots, w_n, t)$ называется *реализацией* функции f. Обозначим через $Th_{1/2}$ класс пороговых функций, для которых существует реализация с $w_1 \geqslant 0, \ldots, w_n \geqslant 0$, $w_1 + \ldots + w_n = 1$, $t = 1/2$. Функции из $Th_{1/2}$ монотонны. Представим функцию $f \in Th_{1/2}$ в виде $f = \bigvee_{\{i_1, \ldots, i_s\}} x_{i_1} \ldots x_{i_s}$, где отсутствуют поглощения конъюнкций (для монотонных функций указанное представление единственно), и сопоставим ей оператор

$$\Theta(r_1, \ldots, r_n) = \bigcup_{\{i_1, \ldots, i_s\}} \bigcap_{i \in \{i_1, \ldots, i_s\}} r_i \bigcap_{j \notin \{i_1, \ldots, i_s\}} r_j^*,$$

который с учетом его содержательных свойств будем называть *оператором взвешенного большинства с правом вето*. Операторы T и Θ до работ автора в научной литературе не встречались.

Будем говорить, что некоторый оператор *однотипен* с $F(r_1, \ldots, r_n)$, если он получается из F заменой каких-либо из отношений r_i на r_i^{-1}. При этом в представлении оператора необходимо заменить отношение r_i на $\overline{r_i^*}$ и r_i^* — на $\bar{r}_i$. Для оператора, однотипного с F, будем использовать обозначение $\hat{F}$. В частности, $\hat{\Lambda}$ означает оператор, однотипный с лексикографией, а $\hat{r}_i$ представляет собой отношение r_i либо r_i^{-1}.

Двойственным оператору F называется оператор F^*, получающийся заменой в представлении F теоретико-множественной операции Φ двойственной операцией

$\Phi^*(Y_1,\dots,Y_n,Z_1,\dots,Z_n)=\bar{\Phi}(\bar{Y}_1,\dots,\bar{Y}_n,\bar{Z}_1,\dots,\bar{Z}_n)$ и отношений r_i двойственными r_i^* (при этом r_i^* заменяются на r_i). Можно проверить, что двойственный оператор Λ^* отличается из Λ лишь тем, что вместо члена $r_1^*\cap\ldots\cap r_{k-1}^*\cap r_k$ в нем присутствует $r_1^*\cap\ldots\cap r_k^*$.

Следующая теорема указывает явный вид всех операторов типа $\mathcal{R}_1^n\to\mathcal{R}_2$, где $\mathcal{R}_1,\mathcal{R}_2\in\{\mathcal{L},\mathcal{W},\mathcal{S},\mathcal{I},\mathcal{P},\mathcal{T},\mathcal{A}\}$, $\mathcal{R}_1\subseteq\mathcal{R}_2$.

Теорема 10 [25–28]. Оператор агрегирования имеет тип $\mathcal{R}_1^n\to\mathcal{R}_2$, $\mathcal{R}_1,\mathcal{R}_2\in\{\mathcal{L},\mathcal{W},\mathcal{S},\mathcal{I},\mathcal{P},\mathcal{T},\mathcal{A}\}$, $\mathcal{R}_1\subseteq\mathcal{R}_2$, тогда и только тогда, когда он может быть представлен в виде, указанном в табл. 1.

Чтобы получить явный вид монотонных операторов типа $\mathcal{R}_1^n\to\mathcal{R}_2$, нужно всюду в табл. 1 заменить $\hat{r}_i$, $\hat{\Lambda}_i$, $\hat{T}_i$ и $\hat{\Theta}_i$ соответственно на r_i, Λ_i, T_i и Θ_i, а произвольные операторы G — монотонными.

Строки и столбцы в таблице отвечают классам $\mathcal{R}_1$ и $\mathcal{R}_2$ соответственно, а в элементах для пар $(\mathcal{R}_1,\mathcal{R}_2)$, таких, что $\mathcal{R}_1\not\subseteq\mathcal{R}_2$, проставлен прочерк. Под G понимается произвольный оператор (остальные обозначения введены выше). Часть представленных результатов, относящаяся к операторам $\mathcal{W}^n\to\mathcal{R}_2$, $\mathcal{R}_2\in\{\mathcal{W},\mathcal{P},\mathcal{T},\mathcal{A}\}$, получена (другими методами) в [5, 24, 29, 30, 31].

Таблица 1

$\mathcal{R}_1$ \ $\mathcal{R}_2$	$\mathcal{L}$	$\mathcal{W}$	$\mathcal{S}$	$\mathcal{I}$	$\mathcal{P}$	$\mathcal{T}$	$\mathcal{A}$
$\mathcal{L}$	$\hat{r}_i$	$\hat{r}_i$	$\hat{r}_i$	$\hat{r}_i$	$\bigcap_i\hat{r}_i$	$\bigcap_i\hat{r}_i\cap\bigcap_j\hat{r}_j^*$	$\hat{r}_i\cap G$
$\mathcal{W}$	–	$\hat{\Lambda}$	$\hat{\Lambda}$	$\hat{\Lambda}$	$\bigcap_i\hat{\Lambda}_i$	$\bigcap_i\hat{\Lambda}_i\cap\bigcap_j\hat{\Lambda}_j^*$	$\hat{\Lambda}\cap G$
$\mathcal{S}$	–	–	$\hat{r}_i$	$\hat{r}_i$	$\bigcap_i\hat{r}_i$	$\bigcap_i\hat{r}_i$	$\hat{\Theta}\cap G$
$\mathcal{I}$	–	–	–	$\hat{r}_i$	$\bigcap_i\hat{r}_i$	$\bigcap_i\hat{r}_i$	$\hat{r}_i\cap G$
$\mathcal{P}$	–	–	–	–	$\bigcap_i\hat{r}_i$	$\bigcap_i\hat{r}_i$	$\hat{r}_i\cap G$
$\mathcal{T}$	–	–	–	–	–	$\bigcap_i\hat{T}_i$	–
$\mathcal{A}$	–	–	–	–	–	–	$\hat{r}_i\cap G$

4.4. Сложность распознавания типов операторов

Теорема описывает способ порождения всех операторов заданного типа, но не дает способа проверки по произвольному оператору, представим ли он в виде, указанном в табл. 1. Будем считать, что теоретико-множественная операция Φ, задающая оператор F, представлена в форме $\cup\cap$ объединения пересечений множеств или их дополнений, либо в форме $\cap\cup$ пересечения объединений множеств или их дополнений. Им соответствуют булевы функции в виде ДНФ либо КНФ. Для монотонной операции Φ существует единственное представление в виде приведенной ДНФ и КНФ (т. е. без отрицаний переменных и поглощений). Будем считать, что монотонная операция типа $\cup\cap$ либо $\cap\cup$ задана приведенной ДНФ либо КНФ, а для представления произвольной операции соответствующего типа может быть взята любая ДНФ или КНФ. Сформулируем результаты о сложности проверки по оператору F (в форме $\cup\cap$ или $\cap\cup$), осуществляет ли он требуемое отображение $\mathcal{R}_1^n\to\mathcal{R}_2$. Ответ на этот вопрос нам не известен лишь для монотонных операторов $\mathcal{S}^n\to\mathcal{A}$ в форме $\cap\cup$.

Теорема 11 [26]. Для пар классов $\mathcal{R}_1,\mathcal{R}_2\in\{\mathcal{L},\mathcal{W},\mathcal{S},\mathcal{I},\mathcal{P},\mathcal{T},\mathcal{A}\}$, $\mathcal{R}_1\subseteq\mathcal{R}_2$, задача выяснения по оператору F, заданному в форме $\cup\cap$ или $\cap\cup$, осуществляет ли он отображение $\mathcal{R}_1^n\to\mathcal{R}_2$, является

а) полиномиальной во всех случаях, когда оператор F монотонен, исключая, быть может, указанный выше случай, где ответ неизвестен;

б) NP-трудной во всех случаях, когда F немонотонен, исключая случаи с $\mathcal{R}_2 = \mathcal{A}$ и F в форме $\cup\cap$, в которых она полиномиальна.

5. Декомпозиция отношений

5.1. Оценки сложности декомпозиций

Будем рассматривать операторы над отношениями, имеющие тот же вид, что и в предыдущем разделе. *Задача декомпозиции отношений* обратна задаче агрегирования и состоит в том, чтобы представить заданное отношение r в виде

$$r = \Phi(r_1, \ldots, r_d, r_1^*, \ldots, r_d^*),$$

где $r_1, \ldots, r_d$ — отношения некоторого класса $\mathcal{R}$; Φ — теоретико-множественная операция. Будем предполагать, что класс $\mathcal{R}$ обеспечивает возможность декомпозиции произвольного отношения r (иррефлексивного). Под *сложностью декомпозиции* будем понимать число d используемых в ней отношений. *Сложность отношения* r (в классе $\mathcal{R}$) будем измерять величиной $d_{\mathcal{R}}(r)$, равной минимальной из сложностей его декомпозиций над $\mathcal{R}$ (использующих произвольные операции Φ). Максимум значений $d_{\mathcal{R}}(r)$ по всем отношениям r на $A^{(n)}$ обозначим $d_{\mathcal{R}}(n)$. Введем аналогичные величины $d^+_{\mathcal{R}}(r)$ и $d^+_{\mathcal{R}}(n)$ для случая, когда в декомпозициях применяются лишь монотонные операции Φ, если класс $\mathcal{R}$ допускает возможность такой декомпозиции для произвольных отношений.

В качестве $\mathcal{R}$ будем использовать классы $\mathcal{L}$, $\mathcal{W}$ и $\mathcal{W}^{(k)}$ отношений линейного порядка, слабого порядка и слабого порядка, имеющих не более k уровней. Напомним [24], что множество A с заданным на нем слабым порядком w разбивается на группы (*уровни*), элементы которых не сравнимы, и уровни могут быть занумерованы так, что xwy тогда и только тогда, когда y принадлежит уровню с большим номером, чем x. Указанный выбор классов для декомпозиции связан с тем, что отношения классов $\mathcal{L}$, $\mathcal{W}$ и $\mathcal{W}^{(k)}$ представимы строгими, нестрогими и k-значными критериями. Эти классы обеспечивают возможность декомпозиции произвольных отношений, в том числе и при использовании лишь монотонных операций Φ [7].

Теорема 12 [7, 32, 33]. Справедливы асимптотические оценки

$$d_{\mathcal{L}}(n) \sim 2\log_2 n,$$
$$2\log_3 n \lesssim d_{\mathcal{W}}(n) \leqslant d_{\mathcal{W}^{(k)}}(n) \leqslant d_{\mathcal{W}^{(2)}}(n) \lesssim 2\log_2(n), \quad 5/3\,\log_2 n \lesssim d_{\mathcal{W}^{(2)}}(n).$$

Величины $d^+_{\mathcal{L}}(n)$, $d^+_{\mathcal{W}}(n)$ и $d^+_{\mathcal{W}^{(k)}}(n)$ асимптотически совпадают с $d_{\mathcal{L}}(n)$, $d_{\mathcal{W}}(n)$ и $d_{\mathcal{W}^{(k)}}(n)$ соответственно.

Ранее в научной литературе исследовалась сложность декомпозиции отношений при использовании специальных типов операций — пересечения [34–36]и мажоритарной [37]. Изложение соответствующих результатов (с доказательствами) имеется в [7].

5.2. Декомпозиции последовательного типа

Этот тип декомпозиций связан с моделью последовательного выбора. Будем говорить, что набор отношений $(r_1, r_2, \ldots, r_k)$ образует *декомпозицию (последовательного типа)* отношения r, если для всех $X \subseteq A$ выполнено

$$C_{r_k}(\ldots C_{r_2}(C_{r_1}(X))\ldots) = C_r(X).$$

В этом случае $r = \Lambda(r_1, r_2, \ldots, r_k)$ [7, 10], где Λ — оператор лексикографии, в представлении которого участвуют все k отношений. Декомпозицию $(r_1, \ldots, r_k)$ назовем *коммутативной*, если любая перестановка этого набора также образует декомпозицию. При этом исходное отношение представимо в виде $r = r_1 \cup \ldots \cup r_k$ [38].

Пусть задан некоторый класс отношений $\mathcal{R}$. Будем рассматривать декомпозиции (последовательного типа), в которых $r_1, \ldots, r_k \in \mathcal{R}$. *Сложность декомпозиции* будем характеризовать числом k входящих в нее отношений, а *сложность отношения* r будем измерять величиной $k_{\mathcal{R}}(r)$ минимальной сложности его декомпозиций в классе $\mathcal{R}$ (если таких декомпозиций нет, $k_{\mathcal{R}}(r)$ не определено). С r свяжем еще одну сложностную характеристику $k^0_{\mathcal{R}}(r)$, определяемую аналогично при дополнительном условии коммутативности декомпозиций. Исследуем вопрос, сколь сильно могут различаться характеристики $k_{\mathcal{R}}(r)$ и $k^0_{\mathcal{R}}(r)$.

Рассмотрим в качестве $\mathcal{R}$ класс $\mathcal{D} = \mathcal{W}^{(2)}$ (отношения из него называются *дихотомиями*). Следующее утверждение показывает, что переход к коммутативным декомпозициям может привести к экспоненциальному росту сложности декомпозиций в сравнении с минимально возможной.

Теорема 13 [38]. Если $r \in \mathcal{W}^{(s)}$, $s \geqslant 2$, то $k_{\mathcal{D}}(r) = \lceil \log_2 s \rceil$, $\quad k^0_{\mathcal{D}}(r) = s - 1$, где $\lceil a \rceil$ означает ближайшее к a целое число, не меньшее a.

Из результатов работы [39] следует, что задача выяснения по произвольному набору отношений $(r_1, r_2, \ldots, r_k)$, образует ли он декомпозицию (последовательного типа) отношения $r = \Lambda(r_1, \ldots, r_k)$, является NP-трудной. В связи с этим представляет интерес изучение отношений r, для которых любое представление $r = \Lambda(r_1, \ldots, r_k)$ дает декомпозицию $(r_1, \ldots, r_k)$. Такие отношения r будем называть *вполне разделимыми*. Будем рассматривать также *коммутативно вполне разделимые отношения* r, для которых всякое представление $r = r_1 \cup \ldots \cup r_k$ дает коммутативную декомпозицию.

Чтобы описать класс всех вполне разделимых отношений, введем некоторые понятия. Скажем, что отношение r', заданное на множестве A', получено из отношения r на множестве A *расщеплением элементов* $x_1, \ldots, x_s \in A$, если $A' = A \cup \{x'_1, \ldots, x'_s\}$ и r' образовано из r добавлением при $i = 1, \ldots, s$ пар (x_i, x'_i), (x'_i, x_i), а также (y, x'_i) и (x'_i, z) для всех таких y и z из A, что yrx_i и x_irz. Если r — отношение и для элемента x нет таких y, что xry, то x называется *минимальным* в r.

Теорема 14 [38].

1. Отношение вполне разделимо тогда и только тогда, когда оно может быть получено из частичного порядка расщеплением некоторых минимальных элементов.

2. Отношение коммутативно вполне разделимо тогда и только тогда, когда оно является частичным порядком.

Из теоремы следует, что в случае асимметричных отношений оба типа вполне разделимости определяют один и тот же класс $\mathcal{P}$ отношений частичного порядка. Теорема 13 показывает, что использование для отношений этого класса коммутативных декомпозиций может привести к экспоненциальному росту сложности.

6. Контекстно-независимые модели

6.1. Понятие контекстно-независимого выбора

Под контекстной независимостью модели выбора понимается отсутствие в описании модели элементов, зависящих от предъявленного множества вариантов. Классические модели выбора, базирующиеся на критериях или отношениях предпочтения,

считаются контекстно-независимыми (КН), поскольку на критериальные оценки или результаты парного сравнения вариантов не влияет состав предъявления. Наличие зависимости от контекста значительно усложняет исследование моделей выбора [8]. Строгого понятия КН-выбора, пригодного для моделей достаточно общего вида, не было, и обсуждения, связанные с контекстной зависимостью либо независимостью, велись на неформальном уровне и касались конкретных моделей. Отсутствие точного определения не позволяло сделать КН-выбор предметом строгого изучения.

Формальное определение КН-выбора дано в [14]. Из этой работы следует, что класс моделей, не зависящих от контекста, включает основные модели, используемые в приложениях. КН-модели допускают единообразное формальное описание и единые методы исследования. Работа [14] в значительной мере основана на результатах из [40], где исследовались КН-модели частного вида — порядковые. Данный раздел написан по материалам [14, 40].

Дальше мы отказываемся от предположения о конечности множества вариантов A — оно может быть произвольным. Предлагаемый подход использует язык узкого исчисления предикатов [41]. Формулы этого языка содержат (наряду с предметными переменными, запятыми и скобками) символы предикатов, логические связки и кванторы, примененные к предметным переменным. Рассматриваются интерпретации формул, в которых предметные переменные принимают значения из множества A и предикаты заданы на всем A. Если $\mathcal{F}(x,\dots,z)$ — формула, всеми свободными переменными которой являются $x,\dots,z$, и если заданы $X \subseteq A$, $\hat{x},\dots,\hat{z} \in X$, то через $\mathcal{F}(\hat{x},\dots,\hat{z})|_X$ будем обозначать значение формулы $\mathcal{F}$ (истина либо ложь) на множестве X при $x=\hat{x},\dots,z=\hat{z}$ (связанные переменные принимают значения из X). Функция выбора C на множестве A называется *контекстно-независимой*, если существует формула $\mathcal{C}(x)$ с одним предметным переменным x такая, что для всех $X \subseteq A$ и $x \in X$ выполнено $x \in C(X) \Leftrightarrow \mathcal{C}(x)|_X$. В этом случае $\mathcal{C}$ называется *формализацией* функции C. Она обеспечивает единое при всех X описание ФВ C (более подробное обсуждение в [14]).

В качестве примеров приведем формализации для трех правил выбора по отношению, введенных в п. 1.3:

$$\mathcal{C}_{r,1}(x) = (\forall y)xr^*y, \quad \mathcal{C}_{r,2}(x) = (\forall y)xry, \quad \mathcal{C}_{r,3}(x) = (\exists y)xry \vee (\forall z)xr^*z,$$

где r^* — двойственное отношение.

6.2. Операции над формализациями

Непосредственное построение формализаций для сколь-нибудь сложных моделей связано со значительными трудностями. Для упрощения этого процесса рассмотрим некоторые операции над КН-функциями и укажем соответствующие преобразования формализаций.

Суперпозиция. Определение этой операции дано в п. 1.4. Чтобы описать формализацию суперпозиции, введем некоторые понятия [42]. Пусть $\mathcal{A}$ — формула узкого исчисления предикатов и P — одноместный предикат. Формула $\mathcal{A}_P$, полученная из $\mathcal{A}$ заменой всех подформул $\forall y\mathcal{F}$ на $\forall y(P(y) \to \mathcal{F})$ и всех подформул $\exists y\mathcal{F}$ на $\exists y(P(y) \wedge \mathcal{F})$, называется *релятивизацией $\mathcal{A}$ относительно P* [42]. Если $\mathcal{C}_1(x)$ и $\mathcal{C}_2(x)$ — формализации ФВ C_1 и C_2, то суперпозиция $C_1 \circ C_2$ имеет формализацию [40]

$$\mathcal{C}(x) = \mathcal{C}_1(x) \wedge \mathcal{C}_{2,\mathcal{C}_1}(x),$$

где $\mathcal{C}_{2,\mathcal{C}_1}$ — релятивизация $\mathcal{C}_2(x)$ относительно $\mathcal{C}_1(x)$.

В качестве примера приведем формализацию функции последовательного выбора $C_{r_1r_2}$ глубины 2, полученную с использованием формализаций для функций C_{r_i} выбора по отношениям, ассоциативности операции суперпозиции и некоторых эквивалентных преобразований:

$$\mathcal{C}_{r_1r_2}(x) = \forall y \exists z (x r_1^* y \wedge (z r_1 y \vee x r_2^* y)).$$

Композиция. Определение этой операции дано в п. 1.5. Если ФВ C образована композицией КН-функций $C_1, \ldots, C_k$ с помощью теоретико-множественной операции F, т. е. $C(X) = F(C_1(X), \ldots, C_k(X))$, то она является КН-функцией и имеет формализацию

$$\mathcal{C}(x) = f(\, \mathcal{C}_1(x), \ldots, \mathcal{C}_k(x)),$$

где f — булева функция, соответствующая операции F.

Ветвление. Пусть Q —некоторое свойство выбора. Будем записывать $Q_X(C)$, если $C(X)$ удовлетворяет свойству Q. *Оператор ветвления* ∇_Q сопоставляет функциям выбора C_1 и C_2 функцию

$$(C_1 \nabla_Q C_2)(X) = \begin{cases} C_1(X), & \text{если } Q_X(C), \\ C_2(X), & \text{если } \bar{Q}_X(C). \end{cases}$$

Свойство Q назовем *формализуемым*, если существует замкнутая формула $\mathcal{Q}(\mathsf{C})$ узкого исчисления предикатов, содержащая переменный одноместный предикат C, и такая, что для любой КН-функции C и ее формализации $\mathcal{C}$ выполнено $\mathcal{Q}(\mathcal{C})|_X \Leftrightarrow Q_X(C)$. Примером может служить свойство «выбирается не более k вариантов». Его формализация имеет вид $\forall x_1 \ldots \forall x_{k+1}(\mathsf{C}(x_1) \wedge \ldots \wedge \mathsf{C}(x_{k+1}) \to \bigvee\limits_{1 \leqslant i < j \leqslant k+1} x_i = x_j)$.

Если C_1 и C_2 — КН-функции и свойство Q формализуемо, то $C = C_1 \nabla_Q C_2$ также является КН-функцией, ибо

$$\mathcal{C}(x) = (\mathcal{C}_1(x) \wedge \mathcal{Q}(\mathcal{C}_1)) \vee (\mathcal{C}_2(x) \wedge \bar{\mathcal{Q}}(\mathcal{C}_1)).$$

Ограниченная обратная связь. Нетрудно видеть, что операция ∇_Q ассоциативна и потому можно рассматривать многоместную операцию $C_1 \nabla_Q C_2 \nabla_Q \ldots \nabla_Q C_s$. При заданном s введем операцию *ограниченной обратной связи*

$$\mathcal{BF}(C_1, C_2, \ldots, C_s \mid Q) = C_1 \nabla_Q C_2 \nabla_Q \ldots \nabla_Q C_s \nabla_Q C^1,$$

где $C^1(X) \equiv X$. Эта операция описывает s-туровую процедуру выбора с обратной связью: выбор $C_i(X)$, произведенный на туре i, считается окончательным, если удовлетворяет свойству Q, в противном случае переходят к туру $i+1$. Если процедура в течение s туров не завершится, она прерывается и возвращается в исходное состояние. Операция $\mathcal{BF}$ в применении к КН-функциям и формализуемому свойству дает КН-функцию, поскольку выражается через ветвление.

Результаты данного пункта могут быть подытожены следующим утверждением.

Теорема 15. Операции суперпозиции, композиции, ветвления и ограниченной обратной связи, примененные к контекстно-независимым функциям выбора, дают контекстно-независимые функции.

Отсюда видно, что большинство используемых в приложениях моделей выбора порождает КН-функции. Иногда с помощью этих операций удается доказать, что модель, исходно построенная как контекстно зависимая, реализует КН-выбор. Примером может служить [14] модель выбора «по числу первых мест», когда выбор осуществляется по некоторой совокупности критериев и выбираются варианты, являющиеся лучшими в данном предъявлении по наибольшему числу критериев.

6.3. Упрощение и анализ формализаций

Будем рассматривать формализации, приведенные к предваренной (пренексной) форме [42]:

$$\mathcal{C}(x) = Q_1 y_1 \dots Q_s y_s \mathcal{F}(x, y_1, \dots, y_s),$$

где $Q_1, \dots, Q_s \in \{\forall, \exists\}$, $\mathcal{F}$ — бесквантворная формула. Такое прдставление будем называть $(Q_1 \dots Q_s)$-*формулой*. Число s кванторов будем называть *рангом формулы*. Ранг характеризует размерность модели выбора. В связи с этим возникает задача минимизации ранга.

Рассмотрим эту задачу применительно к моделям последовательного выбора $C_{r_1 \dots r_k}$ и параллельного выбора $C_{F, r_1 \dots r_k}$. Непосредственное построение формализаций методами п. 6.2 с последующим приведением к предваренной форме дают для этих функций формализации, ранг которых экспоненциально зависит от k. Применение эквивалентных преобразований логики предикатов с использованием ассоциативности операции суперпозиции позволяет существенно уменьшить эту величину.

Теорема 16 [14, 40]. Для последовательного выбора глубины k и параллельного выбора ширины k существуют формализации, ранг которых не превосходит k.

Формализации могут быть использованы для исследования свойств КН-функций. Остановимся на двух свойствах, наиболее часто встречающихся в научной литературе (см., например, [23]). Говорят, что ФВ C обладает свойством

- *наследования*, если $C(X_1 \cup X_2) \subseteq C(X_1) \cup C(X_2)$;
- *согласия*, если $C(X_1 \cup X_2) \supseteq C(X_1) \cap C(X_2)$.

Теорема 17 [40]. Если функция выбора C формализуема посредством (а) $(\forall \dots \forall)$-формулы, (б) $(\forall \exists \dots \exists)$-формулы, то она обладает свойством (а) наследования, (б) согласия.

Рассмотрим примеры применения теоремы. Функция параллельного выбора описывается $(\forall \dots \forall)$-формулой, поэтому она удовлетворяет свойству наследования. Последовательный выбор глубины 2 имеет $(\forall \exists)$-формализацию (см. выше), поэтому для него справедливо свойство согласия. Выбор по отношению описывается $(\forall)$-формулой и, следовательно, удовлетворяет обоим свойствам. Выбор по отношению, основанный на исключении худших вариантов, может быть представлен $(\forall \exists)$-формулой и потому обладает свойством согласия.

Другие применения формализаций для исследования ФВ будут приведены в разделе, относящемся к порядковым моделям выбора.

Часть II. ПОРЯДКОВЫЕ МОДЕЛИ ВЫБОРА

7. Порядковые отношения на $\mathbb{R}^n$

7.1. Порядковые отношения

Будем рассматривать n-мерное действительное пространство $\mathbb{R}^n$, точки которого будем обозначать через $\boldsymbol{x} = (x_1, \dots, x_n)$, $\boldsymbol{y} = (y_1, \dots, y_n)$ и т. д. Для $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ положим

$$\Delta(\boldsymbol{x}, \boldsymbol{y}) = (\operatorname{sgn}(x_1 - y_1), \dots, \operatorname{sgn}(x_n - y_n)),$$

где $\operatorname{sgn} z = -1$, 0 и 1 при $z < 0$, $z = 0$ и $z > 0$ соответственно. Отношение ρ на $\mathbb{R}^n$ называется *порядковым*, если

$$\Delta(\boldsymbol{x}, \boldsymbol{y}) = \Delta(\boldsymbol{x}', \boldsymbol{y}') \Longrightarrow \boldsymbol{x} \rho \boldsymbol{y} \Leftrightarrow \boldsymbol{x}' \rho \boldsymbol{y}'.$$

Если дополнительно выполнено условие $\boldsymbol{x}\rho\boldsymbol{y} \wedge \boldsymbol{z} \geqslant \boldsymbol{x} \Rightarrow \boldsymbol{z}\rho\boldsymbol{y}$, где $\boldsymbol{z} \geqslant \boldsymbol{x} \Leftrightarrow z_1 \geqslant x_1, \ldots, z_n \geqslant x_n$, порядковое отношение (ПО) ρ называется *правильным*.

Порядковые отношения (обычно правильные) часто используются в задачах многокритериального выбора [3]. Точки пространства $\mathbb{R}^n$ интерпретируются как наборы оценок вариантов по n заданным критериям. Хорошо известными примерами правильных ПО являются *отношение Парето* $\boldsymbol{x}\pi\boldsymbol{y} \Longleftrightarrow \boldsymbol{x} \geqslant \boldsymbol{y} \wedge \boldsymbol{x} \neq \boldsymbol{y}$ и *лексикография*

$$\boldsymbol{x}\lambda\boldsymbol{y} \Longleftrightarrow x_1 > y_1 \vee (x_1 = y_1 \wedge x_2 > y_2) \vee \ldots \vee (x_1 = y_1 \wedge \ldots \wedge x_{k-1} = y_{k-1} \wedge x_k > y_k),$$

$k \leqslant n$. При $k = n$ лексикография называется *полной*.

7.2. Логическое описание

Обозначим через $P_{3,2}$ класс двузначных функций $g(u_1, \ldots, u_n) = g(\tilde{u})$ от трехзначных аргументов, $u_s \in \{-1, 0, +1\}$, $1 \leqslant s \leqslant n$, $\forall\tilde{u}(g(\tilde{u}) \in \{0, 1\})$. ПО ρ однозначно задается *представляющей функцией* $g_\rho(\tilde{u})$, связанной с ρ соотношением $\boldsymbol{x}\rho\boldsymbol{y} \Leftrightarrow g_\rho(\Delta(\boldsymbol{x}, \boldsymbol{y})) = 1$. Нетрудно видеть, что отношение ρ правильно тогда и только тогда, когда функция g_ρ монотонна, т. е. удовлетворяет условию $\tilde{u} \geqslant \tilde{v} \Rightarrow g_\rho(\tilde{u}) \geqslant g_\rho(\tilde{v})$.

Введем функции $p(u), p'(u) \in P_{3,2}$ от одного аргумента, положив $p(u) = 1 \Leftrightarrow u > 0$, $p'(u) = 1 \Leftrightarrow u \geqslant 0$. Всякая функция $g \in P_{3,2}$ представима в виде [26]

$$g(u_1, \ldots, u_n) = \varphi(p(u_1), \ldots, p(u_n), p'(u_1), \ldots, p'(u_n)),$$

где φ — булева функция. Введя обозначения p_i, p'_i вместо $p(u_i)$, $p'(u_i)$ и положив $P = (p_1, \ldots, p_n)$, $P' = (p'_1, \ldots, p'_n)$, будем записывать такое представление в виде $g = \varphi(P, P')$. Имея это в виду, иногда будем считать g функцией переменных (P, P') и писать $g(P, P')$.

Выделим два специальных представления функций $g \in P_{3,2}$ — ДНФ и КНФ [26]. С учетом эквивалентных соотношений $p_i \wedge p'_i = p_i$ и $\bar{p}'_i \wedge \bar{p}_i = \bar{p}'_i$ любая конъюнкция $K \not\equiv 0$ от переменных (P, P') может быть приведена к виду $K = q_1 q_2 \ldots q_n$ (значки $\wedge$ конъюнкции опущены), где $q_i \in \{p_i, p'_i, \bar{p}_i, \bar{p}'_i, p'_i\bar{p}_i, 1\}$ ($q_i = 1$ означает отсутствие соответствующего сомножителя). Такие конъюнкции будем называть *элементарными конъюнкциями*, а сомножители q_i — *элементарными сомножителями*. Дизъюнкцию элементарных конъюнкций назовем *дизъюнктивной нормальной формой* (ДНФ). Всякая функция $g \not\equiv 0$ представима в виде ДНФ, а ДНФ функции $g \equiv 0$ считаем равной 0. Любая монотонная функция единственным образом реализуется в виде *приведенной ДНФ*, т. е. ДНФ, не содержащей отрицаний переменных и поглощаемых с учетом соотношений $p_i \leqslant p'_i$ конъюнкций (K_1 поглощает K_2 , если $K_1 \geqslant K_2$). Двойственным образом вводится понятие *элементарной дизъюнкции* $D = d_1 \vee \ldots \vee d_n$, где $d_i \in \{p_i, p'_i, \bar{p}_i, \bar{p}'_i, p_i \vee \bar{p}'_i, 0\}$ называется *элементарным слагаемым*. Всякая функция $g \not\equiv 1$ представима *конъюнктивной нормальной формой* — конъюнкцией элементарных дизъюнкций, а всякая монотонная функция единственным образом представима *приведенной КНФ*, т. е. КНФ, не содержащей отрицаний переменных и поглощаемых дизъюнкций (D_1 поглощает D_2, если $D_1 \leqslant D_2$). Приведенные ДНФ и КНФ функции $g \equiv \mathrm{const}$ считаются совпадающими с этой константой.

Укажем явный вид приведенных ДНФ и КНФ представляющей функции отношения лексикографии:

$$g_\lambda = p_1 \vee p'_1 p_2 \vee p'_1 p'_2 p_3 \vee \ldots \vee p'_1 \ldots p'_{k-1} p_k,$$
$$g_\lambda = p'_1 (p_1 \vee p'_2)(p_1 \vee p_2 \vee p'_3) \ldots (p_1 \vee \ldots \vee p_{k-2} \vee p'_{k-1})(p_1 \vee \ldots \vee p_k).$$

7.3. Операции над отношениями и представляющими функциями

Существенную роль при исследовании свойств ПО играет возможность описания основных операций над ПО в терминах преобразования представляющих функций [26, 43].

1°. Если $\rho = F(\rho_1, \ldots, \rho_k)$, где F — теоретико-множественная операция, $\rho_1, \ldots, \rho_k$ — ПО, то

$$g_\rho(P, P') = \varphi_F(g_{\rho_1}(P, P'), \ldots, g_{\rho_k}(P, P')),$$

где φ_F — булева функция, соответствующая операции F.

Скажем, что ρ' получено из ρ *инвертированием оси* 1, если

$$(x_1, x_2, \ldots, x_n)\rho'(y_1, y_2, \ldots, y_n) \Leftrightarrow (y_1, x_2, \ldots, x_n)\rho(x_1, y_2, \ldots, y_n).$$

Аналогично определяется результат инвертирования оси i. Отношение $\hat{\rho}$, полученное из ρ инвертированием некоторых осей, называется *однотипным* с ρ.

2°. Если ρ' образовано из ПО ρ инвертированием оси i, то представляющая функция $g_{\rho'}$ может быть получена из g_ρ заменой p_i и p'_i соответственно на $\bar{p}'_i$ и $\bar{p}_i$, т. е. при $i = 1$ имеет вид

$$g_{\rho'}(p_1, \ldots, p_n, p'_1, \ldots, p'_n) = g_\rho(\bar{p}'_1, p_2, \ldots, p_n, \bar{p}_1, p'_2, \ldots, p'_n).$$

В случае однотипных отношений указанную замену нужно произвести для всех инвертированных осей.

3°. Представляющая функция отношения ρ^{-1}, обратного к ПО ρ, может быть записана в виде

$$g_{\rho^{-1}}(P, P') = g_\rho(\bar{P}', \bar{P}),$$

где $\bar{P} = (\bar{p}_1, \ldots, \bar{p}_n)$, $\bar{P}' = (\bar{p}'_1, \ldots, \bar{p}'_n)$.

4°. Представляющая функция отношения ρ^*, двойственного к ρ, имеет вид

$$g_{\rho^*}(P, P') = g^*_\rho(P', P),$$

где g^*_ρ — булева функция, двойственная к g_ρ.

Введем операцию композиции $g \circ \hat{g}$ функций $g(P, P')$ и $\hat{g}(P, P')$, которая позволит найти представляющую функцию произведения отношений. Операцию $\circ$ определим вначале для элементарных сомножителей, затем для элементарных конъюнкций и, наконец, — для функций, заданных посредством ДНФ.

Композиция $q_i \circ \hat{q}_i$ элементарных сомножителей находится согласно табл. 2, содержащей в пересечении строки q_i и столбца $\hat{q}_i$ значение $q_i \circ \hat{q}_i$. Композицией элементарных конъюнкций $K = q_1 \ldots q_n$ и $\hat{K} = \hat{q}_1 \ldots \hat{q}_n$ назовем элементарную конъюнкцию

Таблица 2

	p_i	p'_i	$\bar{p}_i$	$\bar{p}'_i$	$p'_i\bar{p}_i$	1
p_i	p_i	p_i	1	1	p_i	1
p'_i	p_i	p'_i	1	1	p'_i	1
$\bar{p}_i$	1	1	$\bar{p}_i$	$\bar{p}'_i$	$\bar{p}_i$	1
$\bar{p}'_i$	1	1	$\bar{p}'_i$	$\bar{p}'_i$	$\bar{p}'_i$	1
$p'_i\bar{p}_i$	p_i	p'_i	$\bar{p}_i$	$\bar{p}'_i$	$p'_i\bar{p}_i$	1
1	1	1	1	1	1	1

$K \circ \hat{K} = (q_1 \circ \hat{q}_1) \dots (q_n \circ \hat{q}_n)$. Композицию функций $g, \hat{g} \not\equiv 0$, заданных посредством ДНФ $g = K_1 \vee \dots \vee K_s$ и $\hat{g} = \hat{K}_1 \vee \dots \vee \hat{K}_t$, определим равенством

$$g \circ \hat{g} = \bigvee_{1 \leqslant u \leqslant s,\, 1 \leqslant v \leqslant t} K_u \circ \hat{K}_v.$$

Теорема 18. Представляющая функция произведения $\rho_1\rho_2$ находится как композиция $g_{\rho_1} \circ g_{\rho_2}$ представляющих функций для ρ_1 и ρ_2.

Если определить *степень* g^k *функции* g, положив $g^1 = g$, $g^k = g^{k-1} \circ g$, $k \geqslant 2$, то последовательность степеней не убывает, т. е. $g \leqslant g^2 \leqslant \dots \leqslant g^k \leqslant \dots$, и ограничена [26]. Результат ее стабилизации обозначим $[g]$.

Произведение порядковых отношений обладает рядом свойств, не имеющих места для произвольных отношений. В их числе коммутативность $\rho_1\rho_2 = \rho_2\rho_1$ (вытекающая из коммутативности операции композиции) и свойство неубывания степеней $\rho \subseteq \rho^2 \subseteq \dots \subseteq \rho^k \subseteq \dots$, где $\rho^1 = \rho$, $\rho^k = \rho^{k-1}\rho$, $k \geqslant 2$. Результат стабилизации этой неубывающей ограниченной последовательности, который обозначим $[\rho]$, является транзитивным замыканием отношения ρ. Легко видеть, что оно является порядковым отношением (а для правильного отношения ρ — правильным) и функция $g_{[\rho]}$ совпадает с $[g_\rho]$.

7.4. Распознавание свойств отношений

Следующая теорема связывает свойства порядковых отношений, определенные в п. 4.2, со свойствами представляющих функций. Пусть $\tilde{0} = (0, \dots, 0)$ и $\tilde{1} = (1, \dots, 1)$ — наборы длины n, $D_0 = p_1 \vee \dots \vee p_n \vee \bar{p}'_1 \vee \dots \vee \bar{p}'_n$.

Теорема 19 [26]. Порядковое отношение ρ (а) рефлексивно, (б) иррефлексивно, (в) асимметрично, (г) антисимметрично, (д) полно, (е) связно, (ж) транзитивно, (з) негатранзитивно, (и) ациклично тогда и только тогда, когда

(а) $g_\rho(\tilde{0}, \tilde{1}) = 1$, (б) $g_\rho(\tilde{0}, \tilde{1}) = 0$, (в) $g_\rho(P, P') \leqslant g^*_\rho(P', P)$, (г) $D_0 g_\rho(P, P') \leqslant$ $\leqslant D_0 g^*_\rho(P', P)$, (д) $g_\rho(P, P') \geqslant g^*_\rho(P', P)$, (е) $D_0 g_\rho(P, P') \geqslant D_0 g^*_\rho(P', P)$, (ж) $g_\rho(P, P') \circ$ $g_\rho(P, P') \leqslant g_\rho(P, P')$, (з) $g^*_\rho(P', P) \circ g^*_\rho(P', P) \leqslant g^*_\rho(P', P)$, (и) $[g_\rho](\tilde{0}, \tilde{1}) = 0$.

Рассмотрим вопрос о сложности распознавания свойств отношений по их представляющим функциям. Будем считать, что представляющие функции заданы посредством ДНФ или КНФ, причем для ПО общего вида эти ДНФ и КНФ произвольны, для правильных ПО являются приведенными. В [26, 43] проведено исследование на NP-трудность и полиномиальность задач распознавания свойств отношений, сформулированных в п. 4.2. Получены следующие результаты.

Теорема 20 [43]. Задачи проверки по представляющим функциям порядковых отношений, заданных в виде ДНФ либо КНФ, обладают ли они свойствами рефлексивности, иррефлексивности, асимметричности, антисимметричности, полноты, связности, транзитивности, негатранзитивности, ацикличности, являются полиномиальными либо NP-трудными в соответствии с тем, как указано в табл. 3, где P и NP означают соответственно полиномиальность и NP-трудность.

7.5. Синтез порядковых отношений с заданными свойствами

Выше логические методы применялись для анализа ПО. Использование этих методов для построения ПО с требуемыми свойствами проиллюстрируем примером отношений с заданным соотношением критериев по важности. Многие алгоритмы построения многокритериальных моделей существенно используют информацию о важности

Таблица 3

Свойство	**Тип отношений**			
	произвольные		правильные	
	Вид задания			
	ДНФ	КНФ	ДНФ	КНФ
Рефлексивность Иррефлексивность	P	P	P	P
Асимметрия Антисимметрия	P	NP	P	NP
Полнота Связность	NP	P	NP	P
Транзитивность Негатранзитивность	NP	NP	P	P
Ацикличность	P	NP	P	P

критериев. Формальные понятия, связанные с важностью (силой) критериев введены в [44, 45] и получили развитие в ряде исследований (см., например, [3]). Дадим соответствующие определения [46], которые несколько отличаются от используемых в указанных работах, поскольку ориентированы на ПО и произвольное соотношение критериев по важности.

Пусть на множестве критериев $\{1,\dots,n\}$ задано рефлексивное отношение r *силы критериев*. Будем считать, что в случаях (а) irj, (б) $irj \wedge j\bar{r}i$, (в) $irj \wedge jri$, (г) $i\bar{r}j \wedge j\bar{r}i$ критерий (а) *не слабее* j, (б) *сильнее* j, критерии i и j (в) *равноценны*, (г) *несравнимы*. Обозначим через $\tilde{u}_{ij}$ набор, полученный из $\tilde{u} \in \{-1,0,+1\}^n$ перестановкой компонент u_i и u_j, $1 \leqslant i,j \leqslant n$. Скажем, что ПО ρ *согласовано с отношением силы* r, если

$$(g_\rho(\tilde{u}) = 1 \wedge irj \wedge (u_i = -1) \wedge (u_j = 1)) \Rightarrow g_\rho(\tilde{u}_{ij}) = 1.$$

Обозначим через τ_r наименьшее транзитивное ПО, согласованное с r и включающее отношение Парето π.

В [46] найден явный вид представляющей функции отношения τ_r для произвольного отношения r силы критериев. Сложность формулы (число символов переменных), задающей представляющую функцию, не превосходит $n(n+1)/2$. Укажем ее для случая, когда r — частичный порядок.

Теорема 21 [46]. Если r — отношение частичного порядка, то

$$g_{\tau_r} = \bigwedge_{1 \leqslant i \leqslant n} \Big(p_i' \vee \bigvee_{j \in r^{-1}(i) \setminus \{i\}} p_j \Big).$$

Рассмотрим важный для приложений частный случй, когда r — слабый порядок. Пусть множество критериев разбито на группы $I_1,\dots,I_k$ и считается, что при $s < t$ критерии из группы I_s сильнее критериев из I_t (а внутри группы они несравнимы). Можно показать, что тогда отношение τ_r приобретает вид $\tau_r = \pi(I_1) \cup \pi(I_1 \cup I_2) \cup \dots \cup \pi(I_1 \cup \dots \cup I_k)$, где через $\pi(I)$ обозначено отношение Парето на множестве критериев $I \subseteq \{1,\dots,n\}$. Отсюда в силу теоремы 14 следует, что выбор по отношению τ_r может быть заменен последовательным выбором по набору отношений Парето. Для паретовского выбора могут быть применены известные эффективные процедуры.

8. Связь порядковых отношений и операторов группового выбора

8.1. Явный вид порядковых отношений

Теорема 19 позволяет найти явный вид порядковых отношений из классов $\mathcal{L}, \mathcal{W}, \mathcal{S}, \mathcal{I}, \mathcal{P}, \mathcal{T}, \mathcal{A}$, определенных в п. 4.2. Отношения, однотипные с лексикографиями и полными лексикографиями, т. е. полученные из них инвертированием некоторых осей, будем называть *обобщенными лексикографиями* и *обобщенными полными лексикографиями*.

Теорема 22 [26, 43]. Порядковое отношение является (а) линейным порядком; (б) слабым порядком, полупорядком, интервальным порядком; (в) частичным порядком; (г) транзитивным отношением; (д) ацикличным отношением тогда и только тогда, когда оно представляет собой (а) обобщенную полную лексикографию; (б) обобщенную лексикографию; (в)пересечение обобщенных лексикографий; (г) пересечение обобщенных лексикографий или пересечение отношений, двойственных обобщенным лексикографиям; (д) может быть дополнено до обобщенной лексикографии.

В случае правильных ПО нужно всюду в формулировке заменить обобщенные и полные обобщенные лексикографии лексикографиями и полными лексикографиями.

Таким образом, для ПО на $\mathbb{R}^n$ классы $\mathcal{W}$, $\mathcal{S}$ и $\mathcal{I}$ совпадают, в то время как в общем случае имеют место строгие включения $\mathcal{W} \subset \mathcal{S} \subset \mathcal{I}$.

Приведем результаты о сложности проверки по представляющим функциям порядковых отношений (произвольных и правильных) в форме ДНФ и КНФ их принадлежности рассматриваемым классам.

Теорема 23 [26, 43]. Задача распознавания по представляющей функции в виде ДНФ или КНФ принадлежности отношения ρ классу $\mathcal{Q} \in \{\mathcal{L}, \mathcal{W}, \mathcal{S}, \mathcal{I}, \mathcal{P}, \mathcal{T}, \mathcal{A}\}$ решается полиномиально для правильных ПО, а для ПО общего вида является NP-трудной во всех случаях, исключая сучай $\mathcal{Q} = \mathcal{A}$ и g_ρ в виде ДНФ, в котором задача решается полиномиально.

8.2. Сведение задачи синтеза операторов группового выбора к распознаванию свойств ПО

Вернемся к задаче синтеза операторов группового выбора для модели свободного выбора, сформулированной и рассмотренной в п. 4.1. Ниже будут приведены результаты, показывающие, что эта задача родственна задаче анализа свойств порядковых отношений, а в некоторых постановках — эквивалентна ей. Найденное соответствие между операторами и порядковыми отношениями позволяет унифицировать исследования в этих областях, ранее проводившиеся параллельно. Оно дает возможность связать ряд результатов по агрегированию с фактами из классической теории отношений (леммой Шпильрайна, теоремой Душника–Миллера) и объяснить наблюдаемые аналогии. Данный подход сводит достаточно сложные объекты (операторы над отношениями) к простым объектам (порядковым отношениям), что облегчает исследование операторов, позволяет использовать геометрическую интуицию. Он предложен в [26], расширен в [25, 27], его изложению и обсуждению посвящен обзор [28]. Связь этого подхода с методом интерпретаций, применяемым в теории моделей, рассмотрен в [47].

Свойства отношений r и классы отношений обычно задаются системами аксиом, которые в большинстве случаев имеют вид

$$\forall x_1 \ldots \forall x_s P(x_1, \ldots, x_s),$$

где P — бесквантворная формула, содержащая (наряду с предметными переменными x_i и логическими операциями) вхождения двуместного предиката $x_i r x_j$ и предиката равенства $x_i = x_j$. Такие свойства и классы отношений будем называть *универсально аксиоматизируемыми*. В частности, универсально аксиоматизируемыми являются все свойства и классы отношений, введенные в п. 4.2 (кванторы общности там опущены и аксиомы записаны в форме $P(x_1, \ldots, x_s)$). Если равенства в аксиомах отсутствуют, то они являются формулами узкого исчисления предикатов [41], и соответствующие свойства и классы отношений будем называть *узко универсально аксиоматизируемыми*. В п. 4.2 такими являются все свойства, исключая антисимметрию и связность, в определении которых участвуют неравенства (отрицания равенств), и все классы, исключая $\mathcal{L}$.

Установим соответствие между опраторами группового выбора рассматриваемого типа $F = \Phi(r_1, \ldots, r_n, r_1^*, \ldots, r_n^*)$ и функциями (3,2)-значной логики, сопоставив оператору F функцию $g_F = \varphi(p_1, \ldots, p_n, p'_1, \ldots, p'_n)$, где φ — булева функция, отвечающая теоретико-множественной операции Φ. Нетрудно видеть, что это соответствие взаимно однозначно. Обозначим через ρ_F ПО с представляющей функцией $g_{\rho_F} = g_F$. Следующая теорема устанавливает связь между операторами F типа $\mathcal{W}^n \to \mathcal{R}$ и соответствующими им ПО ρ_F.

Теорема 24 [26]. Если $\mathcal{R}$ — узко универсально аксиоматизируемый класс отношений, то оператор $F = \Phi(r_1, \ldots, r_n, r_1^*, \ldots, r_n^*)$ осуществляет отображение $\mathcal{W}^n \to \mathcal{R}$ тогда и только тогда, когда соответствующее ему порядковое отношение ρ_F принадлежит $\mathcal{R}$.

Отметим, что теорема не переносится на все универсально (не узко) аксиоматизируемые классы (например, на $\mathcal{L}$). Используя указанное в предыдущем пункте явное описание ПО из основных классов, получаем явный вид операторов типа $\mathcal{W}^n \to \mathcal{R}$, указанный в теореме 10.

Явный вид операторов типа $\mathcal{L}^n \to \mathcal{R}$ может быть получен путем следующего сведения к теореме 24 [27, 28]. Поскольку для отношения $r_i \in \mathcal{L}$ при $x \neq y$ выполнено $x\bar{r}_i y = xr^{-1}y$ и $xr_i y = xr_i^* y$, можно путем эквивалентных преобразований привести оператор $F = \Phi(r_1, \ldots, r_n, r_1^*, \ldots, r_n^*)$ к виду $F = \Psi(r_1, \ldots, r_n, r_1^{-1}, \ldots, r_n^{-1})$, если он порождает иррефлексивные отношения, либо к виду $F = \Psi(r_1^*, \ldots, r_n^*, \bar{r}_1, \ldots, \bar{r}_n)$, если он порождает рефлексивные отношения. Здесь Ψ — монотонная теоретико-множественная операция. Такие операторы будем называть *однородными*, а преобразование, с помощью которого они получены, — *редукцией*.

Теорема 25 [27, 28]. Если $\mathcal{R}$ — узко универсально аксиоматизируемый класс отношений и множество $\mathcal{F}$ операторов решает проблему синтеза для $\mathcal{W}^n \to \mathcal{R}$, то множество всех однородных операторов $\mathcal{F}'$, полученных редукцией операторов $F \in \mathcal{F}$, решает проблему синтеза для $\mathcal{L}^n \to \mathcal{R}$.

Отсюда с учетом легко проверяемого факта, что редукция обобщенной лексикографии дает оператор r_i либо r_i^{-1}, получаем явный вид операторов $\mathcal{L}^n \to \mathcal{R}$, указанный в теореме 10.

Для других классов отношений $\mathcal{R}_1$ не удалось сформулировать общих результатов о виде операторов $\mathcal{R}_1^n \to \mathcal{R}_2$, подобных теоремам для $\mathcal{W}$ и $\mathcal{L}$. Однако сам подход к описанию операторов $\mathcal{R}_1^n \to \mathcal{R}_2$ на основе представляющих функций g_ρ отношений $\rho \in \mathcal{R}_2$ применим (в некотором модифицированном виде) и для других классов отношений [25, 26]. Этим способом был установлен явный вид всех операторов в теореме 10. Наибольшие трудности вызвали операторы $\mathcal{S}^n \to \mathcal{A}$ и $\mathcal{T}^n \to \mathcal{T}$.

9. Порядковые отношения в произвольных критериальных пространствах

9.1. Критериальные пространства общего вида

Выше рассматривались порядковые отношения на $\mathbb{R}^n$. Но в прикладных задачах выбора обычно применяются критерии с ограниченным множеством значений — качественные, балльные, количественные с конечными шкалами. Отношения на $\mathbb{R}^n$ обладают рядом полезных свойств, облегчающих их исследование. Так, например, произведение порядковых отношений в пространстве $\mathbb{R}^n$ является порядковым, а в пространствах с конечными шкалами нет. Доказательства для пространства $\mathbb{R}^n$ используют его плотность и на общий случай пространств не переносятся, некоторые результаты приобретают другой вид. Данный раздел посвящен ПО в произвольных критериальных пространствах.

Будем рассматривать критериальные пространства $\mathbb{X} = X_1 \times \cdots \times X_n$, являющиеся декартовыми произведениями некоторых подмножеств X_i множества $\mathbb{R}$ действительных чисел. Упорядоченное множество X_i будем называть *шкалой* критерия i, а его мощность $|X_i|$, обозначаемую дальше $\varkappa_i$, — *значностью* критерия i (шкалы i). Критерии могут быть конечнозначными и бесконечнозначными. Минимальную из значностей $\varkappa_i$ шкал, образующих пространство $\mathbb{X}$, обозначим через $\varkappa(\mathbb{X})$ и назовем *индексом пространства* $\mathbb{X}$. Порядковые отношения на $\mathbb{X}$ и связанные с ними понятия определяются как раньше, если всюду в определениях заменить $\mathbb{R}^n$ на $\mathbb{X}$.

Дальше будем считать, что $\varkappa(\mathbb{X}) \geqslant 2$. Тогда множество всех $\Delta(\boldsymbol{x}, \boldsymbol{y})$ для $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{X}$ совпадает с $\{-1, 0, 1\}^n$, и отношение однозначно распространяется на любые $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. Отношение на всем $\mathbb{R}^n$, определяемое 3^n значениями $\Delta(\boldsymbol{x}, \boldsymbol{y})$, будем называть *пополнением* исходного отношения. Для пополнения будем использовать обозначение ρ, а для исходного отношения, которое можно рассматривать как сужение ρ на $\mathbb{X}$, — обозначение $\rho_{\mathbb{X}}$. Представляющей функцией отношения $\rho_{\mathbb{X}}$ будем считать представляющую функцию g_ρ его пополнения.

9.2. Вложение отношений в критериальные пространства

Рассмотрим задачу вложения отношений, заданных на произвольном конечном множестве A, в критериальные пространства. Это позволяет заменять модели свободного выбора моделями многокритериального выбора по порядковым отношениям. Поскольку трудоемкость многокритериального выбора существенно зависит от числа критериев, ставится задача вложения в пространства возможно меньшей размерности. Дадим соответствующие определения.

Пусть r — отношение на конечном множестве A. Будем говорить, что r *вложимо в критериальное пространство* $\mathbb{X}$, если можно так приписать вариантам $x \in A$ наборы $\boldsymbol{x} \in \mathbb{X}$ и назначить такое порядковое отношение ρ на $\mathbb{X}$, что $xry \Leftrightarrow \boldsymbol{x}\rho\boldsymbol{y}$ для любых $x, y \in A$. В случае правильных порядковых отношений будем говорить о *правильном вложении*. Обозначим через $\mathbb{X}_A$ множество наборов $\boldsymbol{x} \in \mathbb{X}$, сопоставленных при вложении вариантам $x \in A$. Отношения r будем предполагать иррефлексивными (для рефлексивных результаты аналогичны).

Выделим некоторые типы вложений. Критерий i назовем *строгим*, если для любых $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{X}_A$, $\boldsymbol{x} \neq \boldsymbol{y}$, имеет место $x_i \neq y_i$, а в противном случае — *нестрогим*. Если условие строгости выполнено при всех i, будем говорить о вложении в пространство строгих критериев. В случае, когда все критерии являются нестрогими и принимают не более k значений, будем говорить о вложении в пространство k-значных критериев. Следующий результат устанавливает связь между вложениями в

критериальные пространства и декомпозициями отношений, рассмотренными в п. 5.1. Под Ф понимается теоретико-множественная операция.

Теорема 26 [7, 32, 33]. Отношение r вложимо в d-мерное пространство с (а) произвольными, (б) k-значными, (в) строгими критериями тогда и только тогда, когда оно представимо в виде

$$r = \Phi(r_1, \ldots, r_d, r_1^*, \ldots, r_d^*),$$

где отношения $r_1, \ldots, r_d$ принадлежат (а) классу $\mathcal{W}$, (б) классу $\mathcal{W}^{(k)}$, (в) классу $\mathcal{L}$.

Эти результаты переносятся на правильные вложения, если в качестве операции Ф взять монотонную.

Замечание. Поскольку для линейных порядков r_i при $x \neq y$ выполнено $xr_i^*y = xr_iy$, то в случае строгих критериев представление, указанное в теореме, сводится к более простому $r = \Phi(r_1, \ldots, r_d)$.

Всякое отношение r вложимо и правильно вложимо при всех рассматриваемых типах вложения в критериальные пространства подходящей размерности [7]. Обозначим через $d(r)$, $\hat{d}(r)$ и $d_k(r)$ минимальные размерности пространств с соответственно произвольными, строгими и k-значными критериями, в которые вложимо r, а через $d(n)$, $\hat{d}(n)$ и $d_k(n)$ — их максимальные значения по всем отношениям (иррефлексивным) на множестве A мощности n. Аналогичные величины $d^+(n)$, $\hat{d}^+(n)$ и $d_k^+(n)$ введем для правильных вложений. Из теоремы 12 в силу предшествующей теоремы получаем следующие результаты.

Теорема 27 [7, 32, 33]. Справедливы асимптотические оценки

$$\hat{d}(n) \sim 2\log_2 n,$$
$$2\log_3 n \lesssim d(n) \leqslant d_k(n) \leqslant d_2(n) \lesssim 2\log_2(n), \quad 5/3\ \log_2 n \lesssim d_2(n).$$

Величины $\hat{d}^+(n)$, $d^+(n)$ и $d_k^+(n)$ асимптотически совпадают с $\hat{d}(n)$, $d(n)$ и $d_k(n)$ соответственно.

9.3. Сведение к отношениям на $\mathbb{R}^n$

В данном пункте описан подход, развитый в [48, 49], который сводит исследование свойств порядковых отношений в пространствах с конечными шкалами к аналогичным задачам для $\mathbb{R}^n$ и позволяет воспользоваться многими результатами, полученными для $\mathbb{R}^n$. Он родствен некоторым подходам, применяемым в математической логике и теории моделей. Формулировка этого подхода в терминах теории моделей имеется в [50].

Будем рассматривать универсально аксиоматизируемые свойства отношений (см. п. 8.2). Такие свойства задаются аксиомами, использующими лишь кванторы общности. *Рангом аксиомы* называется число s участвующих в ней кванторов. *Ранг системы аксиом* считается равным максимальному из рангов входящих в нее аксиом, если система конечна, и полагается равным ω в случае бесконечной системы. *Рангом универсально аксиоматизируемого свойства* называется минимальный из рангов задающих его систем аксиом.

Пусть на множестве пар $(x, y) \in A^2$ задано отношение эквивалентности $\sim$. Эквивалентность $\sim$ называется *строгой*, если из $(x, y) \sim (x', y')$ и $x = y$ следует $x' = y'$. Дальше будем рассматривать только строгие эквивалентности. Множество X, $X \subseteq A$, называется *s-представительным* (для A относительно эквивалентности $\sim$), если для любых $x_1, \ldots, x_s \in A$ найдутся $x'_1, \ldots, x'_s \in X$ такие, что $(x'_i, x'_j) \sim (x_i, x_j)$, $1 \leqslant i, j \leqslant s$. Множество X, s-представительное при любом $s \geqslant 1$, называется *ω-представительным*.

Эквивалентность $\sim$ задает класс $\mathcal{K}_\sim$ отношений r на A, удовлетворяющих условию $(x,y) \sim (x',y') \Longrightarrow xry \Leftrightarrow x'ry'$. Предлагаемый подход основан на следующем утверждении, в котором для $X \subseteq A$ через r_X обозначено отношение $r \cap X^2$.

Теорема 28 [49]. Если Φ — универсально аксиоматизируемое свойство ранга не выше s, $s \in N \cup \{\omega\}$, множество X является s-представительным для A и r — отношение класса $\mathcal{K}_\sim$, то отношение r_X обладает свойством Φ тогда и только тогда, когда им обладает r.

Класс порядковых отношений может рассматриваться как $\mathcal{K}_\sim$, если положить $A = \mathbb{R}^n$ и в качестве отношения эквивалентности взять

$$(\boldsymbol{x},\boldsymbol{y}) \sim (\boldsymbol{x'},\boldsymbol{y'}) \Leftrightarrow \Delta(\boldsymbol{x},\boldsymbol{y}) = \Delta(\boldsymbol{x'},\boldsymbol{y'}).$$

Обозначим через E_k^n декартову степень множества $E_k = \{0,1,\ldots,k-1\}$. Допускается $k = \omega$, где $E_\omega = \{0,1,2,\ldots\} = N$. Нетрудно доказать, что при любом $s \in N \cup \{\omega\}$ множество E_s^n является s-представительным для $\mathbb{R}^n$ по введенному выше отношению эквивалентности. Если $\varkappa(\mathbb{X}) = s$, то можно считать, что $\mathbb{X} \supseteq E_s^n$. Это позволяет переформулировать теорему 28 для порядковых отношений на $\mathbb{X}$.

Теорема 29. Если ранг универсально аксиоматизируемого свойства Φ не превышает s, $s \in N \cup \{\omega\}$, а индекс $\varkappa(\mathbb{X})$ пространства $\mathbb{X}$ не ниже s, то порядковое отношение ρ на $\mathbb{X}$ обладает свойством Φ тогда и только тогда, когда им обладает его пополнение.

Подсчитав ранги введенных в п. 8.2 свойств отношений, получаем следующее утверждение.

Следствие 1. Порядковое отношение ρ на $\mathbb{X}$ обладает одним из свойств рефлексивности, иррефлексивности, асимметрии, антисимметрии, полноты, связности тогда и только тогда, когда соответствующим свойством обладает его пополнение. В случае $\varkappa(\mathbb{X}) \geqslant 3$ аналогичное утверждение имеет место и для свойств транзитивности и негатранзитивности.

9.4. Сравнение свойств порядковых отношений на $\mathbb{R}^n$ и в дискретных пространствах

Описанный подход позволяет перенести большинство результатов, методов и алгоритмов, развитых для $\mathbb{R}^n$, на произвольные пространства $\mathbb{X}$ с $\varkappa(\mathbb{X}) \geqslant 3$. Остановимся в большей мере на возникающих здесь отличиях. Данный пункт написан по материалам [48, 49].

Будем рассматривать те же операции над ПО, что и в п. 7.3 — теоретико-множественные, инвертирование осей, обращение отношений, переход к двойственным отношениям, произведение отношений. Преобразования представляющих функций $1^\circ - 4^\circ$ из п. 7.3, относящиеся к первым четырем операциям, остаются справедливыми для ПО на произвольном $\mathbb{X}$.

Если $\mathbb{X}$ отлично от $\mathbb{R}^n$, то произведение порядковых отношений не обязательно будет ПО. Для иллюстрации рассмотрим одномерное ПО $>$ («больше») на N. Его произведение само на себя не является порядковым, так как $(2,0) \in (> \cdot >)$, а $(1,0) \notin (> \cdot >)$. В дальнейшем роль произведения $[\rho_1\rho_2]$ будет играть его *порядковое замыкание* $[\rho_1\rho_2]$. Под порядковым замыканием $[\rho]$ отношения ρ (не обязательно порядкового) на $\mathbb{X}$ понимается наименьшее по включению ПО на $\mathbb{X}$, содержащее ρ.

Операция композиции $g \circ \hat{g}$ функций g и $\hat{g}$, позволяющая найти представляющую функцию для порядкового замыкания произведения отношений, определяется, как и

в п. 7.3, с тем отличием, что при нахождении композиции $q_i \circ \hat{q}_i$ элементарных сомножителей учитывается значность $\varkappa_i$ критериев i. Если $\varkappa_i \geqslant 3$ (в частности, в случае бесконечной значности), композиция $q_i \circ \hat{q}_i$ находится, как в п. 7.3 (см. табл. 2), если $\varkappa_i = 2$, то в соответствии с табл. 4.

Таблица 4

	p_i	p'_i	$\bar{p}_i$	$\bar{p}'_i$	$p'_i\bar{p}_i$	1
p_i	0	p_i	p'_i	$p'_i\bar{p}_i$	p_i	p'_i
p'_i	p_i	p'_i	1	$\bar{p}_i$	p'_i	1
$\bar{p}_i$	p'_i	1	$\bar{p}_i$	$\bar{p}'_i$	$\bar{p}_i$	1
$\bar{p}'_i$	$p'_i\bar{p}_i$	$\bar{p}_i$	$\bar{p}'_i$	0	$\bar{p}'_i$	$\bar{p}_i$
$p'_i\bar{p}_i$	p_i	p'_i	$\bar{p}_i$	$\bar{p}'_i$	$p'_i\bar{p}_i$	1
1	p'_i	1	1	$\bar{p}_i$	1	1

Теорема 18 приобретает для общего случая критериальных пространств следующий вид.

Теорема 30. Для порядковых отношений на произвольном $\mathbb{X}$ представляющая функция порядкового замыкания $[\rho_1 \cdot \rho_2]$ произведения ПО находится как композиция $g_{\rho_1} \circ g_{\rho_2}$ представляющих функций для ρ_1 и ρ_2.

Указанные факты позволяют распространить на произвольные критериальные пространства $\mathbb{X}$ установленные в теореме 19 для ПО на $\mathbb{R}^n$ соотношения между свойствами ПО и представляющими функциями. Это относится ко всем свойствам, исключая ацикличность, которая имеет бесконечный ранг и выпадает из общего рассмотрения. Ацикличные ПО на $\mathbb{R}^n$ и в дискретных пространствах ведут себя по-разному.

Результаты теоремы 20 о сложности распознавания свойств ПО на $\mathbb{R}^n$, относящиеся к свойствам рефлексивности, иррефлексивности, асимметрии, антисимметрии, полноты и связности переносятся на пространства $\mathbb{X}$ общего вида, а для свойств транзитивности и негатранзитивности — на пространства с $\varkappa(\mathbb{X}) \geqslant 3$. Условие $\varkappa(\mathbb{X}) \geqslant 3$ связано с тем, что ранг двух последних свойств равен 3. При $\varkappa(\mathbb{X}) \geqslant 3$ на пространства $\mathbb{X}$ распространяются установленные в теореме 22 для пространства $\mathbb{R}^n$ результаты о явном виде ПО из всех основных классов, исключая результаты для класса $\mathcal{A}$ ацикличных отношений. При этом для классов $\mathcal{L}$, $\mathcal{W}$, $\mathcal{P}$ и $\mathcal{T}$, задаваемых свойствами ранга 3, достаточно применить теорему 29, а классы $\mathcal{S}$ и $\mathcal{I}$, в определении которых присутствуют свойства ранга 4, требуют специального рассмотрения. Из сказанного и теоремы 23 следует, что при $\varkappa(\mathbb{X}) \geqslant 3$ сложность проверки по представляющей функции в форме ДНФ либо КНФ принадлежности ПО на множестве $\mathbb{X}$ каждому из введенных в п. 4.2 классов, исключая $\mathcal{A}$, полиномиальна для правильных ПО и NP-трудна для произвольных ПО.

Условие $\varkappa(\mathbb{X}) \geqslant 3$, при котором установлены приведенные выше результаты о явном виде отношений, является существенным. В [49] показано, что в случае двузначных критериев каждый из рассматриваемых классов отношений шире, чем дает описание. Кроме того, в [49] установлено, что при использовании двузначных критериев классы $\mathcal{W}$, $\mathcal{S}$ и $\mathcal{I}$ попарно различны, в то время как в случае $\varkappa(\mathbb{X}) \geqslant 3$ в соответствии с теоремой 23 они совпадают.

Как уже говорилось, свойство ацикличности, имеющее бесконечный ранг, стоит особняком. Ни при каком конечном s ацикличность ПО в пространстве s-значных

критериев не гарантирует его ацикличности на $\mathbb{R}^n$. Следующая теорема указывает, в каких границах описанный подход применим к свойству ацикличности.

Теорема 31. Если $\mathbb{X}$ — пространство размерности n и индекса $\varkappa(\mathbb{X}) = s$, то при $n \leqslant s$ порядковое отношение на $\mathbb{X}$ ациклично тогда и только тогда, когда оно ациклично на $\mathbb{R}^n$. При $n > s$ существуют порядковые отношения, ацикличные на E^n_s и имеющие циклы в $\mathbb{R}^n$.

Приведем результат, показывающий, что свойства ацикличных отношений в пространствах с конечнозначными критериями существенно отличаются от свойств ацикличных отношений на $\mathbb{R}^n$. Согласно теореме 22, всякое ацикличное ПО на $\mathbb{R}^n$ может быть дополнено до линейного (и, следовательно, частичного) порядка.

Теорема 32. При любых s и n, $s \geqslant 2$, $n > s$ в E^n_s существует ацикличное отношение, не дополнимое до частичного порядка.

10. Порядковые модели

10.1. Формализация

Данный раздел написан по материалам [14, 40]. Будем рассматривать ФВ на $\mathbb{R}^n$. Отображение $\psi : \mathbb{R}^n \to \mathbb{R}^n$ назовем *порядковым отображением*, если для любых $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ имеет место $\Delta(\boldsymbol{x}, \boldsymbol{y}) = \Delta(\psi(\boldsymbol{x}), \psi(\boldsymbol{y}))$. Легко видеть, что отображение $\mathbb{R}^n \to \mathbb{R}^n$ порядково тогда и только тогда, когда при каждом i, $1 \leqslant i \leqslant n$, производит строго монотонное преобразование i-й координаты точек из $\mathbb{R}^n$.

В терминах порядковых отображений можно дать другое определение порядкового отношения, эквивалентное введенному ранее. ПО ρ на $\mathbb{R}^n$ — это отношение, которое для любых $\boldsymbol{x}, \boldsymbol{y}$ и порядкового отображения ψ удовлетворяет соотношению $\psi(\boldsymbol{x})\rho\psi(\boldsymbol{y}) \Leftrightarrow \boldsymbol{x}\rho\boldsymbol{y}$. Этот подход может быть распространен на функции выбора. ФВ C на $\mathbb{R}^n$ называется *порядковой функцией* [3], если для любых $X \subseteq \mathbb{R}^n$ и порядкового отображения ψ выполнено $C(\psi(X)) = \psi(C(X))$, где $\psi(Z)$ — образ множества Z. Модель выбора M назовем *порядковой моделью*, если C_M — порядковая ФВ.

Будем рассматривать контекстно-независимые порядковые ФВ и для их формализации будем применять способ, описанный в п. 6.1. Уточним этот способ применительно к порядковым ФВ.

Будем рассматривать интерпретации, в которых значениями предметных переменных являются точки $\boldsymbol{x} = (x_1, \ldots, x_n)$ пространства $\mathbb{R}^n$. Предикат $P(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ на $\mathbb{R}^n$ будем называть *порядковым предикатом*, если для любых $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ и порядкового отображения ψ выполнено $P(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \Leftrightarrow P(\psi(\boldsymbol{x}_1), \ldots, \psi(\boldsymbol{x}_k))$. Формулу (узкого исчисления предикатов) назовем *порядковой формулой*, если в ней используются лишь порядковые предикаты. Порядковая ФВ C называется *формализуемой*, если существует такая порядковая формула $\mathcal{C}(\boldsymbol{x})$ (*формализация*), что для любых $X \subseteq \mathbb{R}^n$ и $\boldsymbol{x} \in X$ выполнено $\boldsymbol{x} \in C(X) \Leftrightarrow \mathcal{C}(\boldsymbol{x})|_X$, где $\mathcal{C}(\boldsymbol{x})|_X$ — значение формулы на множестве X (см. п. 6.1).

Если для обозначения значений формул вместо И («истина») и Л («ложь») использовать 1 и 0, то всякий порядковый предикат P может быть представлен в виде $P(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) = g(\Delta(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{j_1}), \ldots, \Delta(\boldsymbol{x}_{i_r}, \boldsymbol{x}_{j_r}))$, где $g \in P_{3,2}$ [40, 14]. Записав формулу $\mathcal{C}(\boldsymbol{x})$ в предваренной форме

$$\mathcal{C}(\boldsymbol{x}) = Q_1\boldsymbol{y}_1 \ldots Q_s\boldsymbol{y}_s\mathcal{F}(\boldsymbol{x}, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_s)$$

и использовав указанное выше представление для порядкового предиката, реализуемого бесквантормой формулой $\mathcal{F}$, получим

$$\mathcal{C}(\boldsymbol{x}) = Q_1\boldsymbol{y}_1 \dots Q_s\boldsymbol{y}_s g(\Delta(\boldsymbol{z}_{i_1}, \boldsymbol{z}_{j_1}), \dots, \Delta(\boldsymbol{z}_{i_r}, \boldsymbol{z}_{j_r})),$$

где $g \in P_{3,2}$, $\boldsymbol{z}_{i_1}, \boldsymbol{z}_{j_1}, \dots, \boldsymbol{z}_{i_r}, \boldsymbol{z}_{j_r} \in \{\boldsymbol{x}, \boldsymbol{y}_1, \dots, \boldsymbol{y}_s\}$. Это выражение будем называть *формализацией в стандартной форме*.

10.2. Построение формализаций и доказательство неформализуемости

В порядковых моделях в качестве элементарных актов выбора обычно используется выбор по порядковому отношению. На этой основе строятся более сложные модели.

В [40] проведено исследование на формализуемость встречающихся в литературе правил выбора по отношению (порядковому). Для каждого из них построена формализация либо доказана неформализуемость. В абсолютном большинстве своем правила оказались формализуемыми. В частности, формализуемы правила, введенные в п. 1.3. Первое из них, например, имеет формализацию $\mathcal{C}_\rho(\boldsymbol{x}) = \forall\boldsymbol{y} g^*_\rho(\Delta(\boldsymbol{x}, \boldsymbol{y}))$, где g_ρ — представляющая функция, а $*$ означает двойственность. К неформализуемым относится турнирное правило Коупленда [8].

Для построения более сложных моделей выбора используются некоторые операции над моделями. Они могут быть описаны в терминах операций над соответствующими функциями выбора. Как и в п. 6.2, будем рассматривать операции композиции, суперпозиции, ветвления и ограниченной обратной связи. На свойство Q, используемое в двух последних операциях, наложим условие, чтобы оно было порядковым, т. е. представлялось формулой, содержащей лишь порядковые предикаты. Теорема 15 распространяется на порядковые модели: применение указанных операций к формализуемым порядковым моделям дает формализуемые порядковые модели. Их формализация строится, как в п. 6.2. Так, например, формализация выбора глубины два по ПО ρ_1 и ρ_2 имеет вид

$$\mathcal{C}_{\rho_1\rho_2}(\boldsymbol{x}) = \forall\boldsymbol{y}\exists\boldsymbol{z}(g^*_{\rho_1}(\Delta(\boldsymbol{x}, \boldsymbol{y}) \wedge (g_{\rho_1}(\Delta(\boldsymbol{z}, \boldsymbol{y}) \vee g^*_{\rho_2}(\Delta(\boldsymbol{x}, \boldsymbol{y}))).$$

Возможность представления формализаций для порядковых моделей в стандартной форме облегчает доказательство неформализуемости и, следовательно, — контекстной зависимости порядковых ФВ. Так, в [14] методом из [40] доказана контекстная зависимость выбора для известного правила Борда выбора по сумме рангов [8], а в [40] — для правила Коупленда и ряда других правил выбора по ПО. Для ФВ общего вида (не порядковых) нет ограничений на вид формализаций, и доказать контекстную зависимость для каких-либо ФВ общего вида пока не удалось.

10.3. Упрощение формализаций

На порядковые ФВ переносятся результаты об упрощении формализаций для контекстно-независимых функций, приведенные в п. 6.3. В частности, модели последовательного выбора глубины k и параллельного выбора ширины k по порядковым отношениям допускают формализацию в стандартной форме, имеющую ранг не выше k. Но для порядковых моделей удается поставить и решить более широкий круг задач, чем для КН-моделей общего типа. Одной из них является задача построения для моделей параллельного выбора формализаций, имеющих наименьший ранг. Изложенные ниже результаты взяты из [40].

Формулу вида

$$\mathcal{C}(\boldsymbol{x}) = \forall\boldsymbol{y}_1 \dots \forall\boldsymbol{y}_s g(\Delta(\boldsymbol{x}, \boldsymbol{y}_1), \dots, \Delta(\boldsymbol{x}, \boldsymbol{y}_s)), \quad g \in P_{3,2},$$

будем называть *однородной* $\forall$*-формулой*, а в случае монотонной функции g — *монотонной однородной* $\forall$*-формулой*. Порядковая ФВ представима моделью параллельного выбора тогда и только тогда, когда она допускает формализацию однородной $\forall$-формулой с функцией g, удовлетворяющей условию $g(0,\dots,0)=1$. Аналогичный факт справедлив для моделей параллельного выбора, использующих правильные ПО, если добавить условие монотонности однородных $\forall$-формул. Как и раньше, формулу будем характеризовать ее рангом — числом кванторов. В связи с моделью параллельного выбора возникает задача минимизации однородных и монотонных однородных $\forall$-формул, состоящая в том, чтобы по однородной (монотонной однородной) $\forall$-формуле с $g(0,\dots,0)=1$ построить формулу того же типа, имеющую наименьший ранг. Функция g предполагается заданной посредством КНФ, а в монотонном случае — посредством приведенной КНФ. Получены следующие результаты.

Теорема 33.

1. Задача минимизации однородных $\forall$-формул алгоритмически разрешима, но NP-трудна.

2. Задача минимизации монотонных однородных $\forall$-формул полиномиальна.

Отметим, что указанный в теореме полиномиальный алгоритм, помимо того, что обеспечивает формулы с наименьшим числом кванторов, дает представление участвующей в них функции из $P_{3,2}$ в виде КНФ наименьшей сложности, т. е. с наименьшим числом вхождений символов переменных.

11. Синтез и аппроксимация

Данный раздел написан по материалам [40]. Будем рассматривать задачи синтеза и аппроксимации в классе порядковых отношений. Дальше, говоря о синтезе и аппроксимациях, будем иметь в виду этот класс моделей. Задача синтеза состоит в том, чтобы по формализации $\mathcal{C}(\mathbf{x})$ порядковой ФВ C построить реализующее ее ПО либо установить, что это невозможно. Аналогично ставятся задачи о верхней и нижней аппроксимациях в классе ПО. Следующий результат показывает, что при решении этих задач возникают принципиальные трудности.

Теорема 34. Задачи синтеза и построения верхней и нижней аппроксимаций в классе порядковых отношений являются алгоритмически неразрешимыми.

Трудности, связанные с неразрешимостью указанных проблем, частично удается преодолеть с помощью парно-выявленных отношений (это понятие введено в п. 2.2). Будем рассматривать порядковые ФВ, удовлетворяющие условию $C(\{\boldsymbol{x}\})=\{\boldsymbol{x}\}$ (выбор в одноэлементных предъявлениях непуст). Для них парно-выявленное отношение $\hat\rho$, которое определяется посредством соотношения $\boldsymbol{x}\hat\rho\boldsymbol{y}\Leftrightarrow\boldsymbol{y}\notin C(\{\boldsymbol{x},\boldsymbol{y}\})$, будет порядковым. Ясно, что $C_{\hat\rho}(\{\boldsymbol{x},\boldsymbol{y}\})=C(\{\boldsymbol{x},\boldsymbol{y}\})$.

В [40] предложен полиномиальный (относительно размерности n пространства при фиксированном s) способ построения парно-выявленного отношения $\hat\rho$ по формализации $\mathcal{C}(\boldsymbol{x})$ ранга s в стандартной форме. Операциям над ФВ, рассмотренным в п. 6.2, соответствуют определенные преобразования парно-выявленных отношений. Приведем преобразования, соответствующие операциям композиции и суперпозиции (для других операций см. в [40]). Обозначим через $\hat\rho,\hat\rho_1,\dots,\hat\rho_k$ парно-выявленные отношения для функций $C,C_1,\dots,C_k$.

Теорема 35.

1. Если ФВ C образована композицией $C=F(C_1,\dots,C_k)$, то $\hat\rho=F^*(\hat\rho_1,\dots,\hat\rho_k)$, где F^* — двойственная к F теоретико-множественная операция.

2. Если ФВ C образована суперпозицией $C = C_1 \circ C_2$, то $\hat{\rho} = \hat{\rho}_1 \cup (\hat{\rho}_1 \cap \hat{\rho}_2)$.

Следствие 2.

1. Парно-выявленное отношение функции $C_{F,\rho_1,\ldots,\rho_k}$ параллельного выбора имеет вид $\hat{\rho} = F^*(\hat{\rho}_1, \ldots, \hat{\rho}_k)$.

2. Парно-выявленное отношение функции $C_{\rho_1,\ldots,\rho_k}$ последовательного выбора имеет вид $\hat{\rho} = \hat{\rho}_1 \cup (\hat{\rho}_1 \cap \hat{\rho}_2) \cup \ldots \cup (\hat{\rho}_1 \cap \ldots \cap \hat{\rho}_k)$.

Возможность конструктивного нахождения парно-выявленных отношений для формализуемых ФВ облегчает решение ряда задач, связанных с построением моделей. Приведем некоторые относящиеся сюда утверждения.

1°. Если ФВ C представима отношением, то этим отношением является парно-выявленное отношение $\hat{\rho}$.

2°. Если для ФВ C выполнено условие $C \subseteq C_{\hat{\rho}}$ ($C \supseteq C_{\hat{\rho}}$), то парно-выявленное отношение $\hat{\rho}$ реализует верхнюю (нижнюю) аппроксимацию функции C.

3°. Если ФВ C удовлетворяет свойству наследования (согласия), то парно-выявленное отношение $\hat{\rho}$ реализует верхнюю (нижнюю) аппроксимацию функции C.

4°. Парно-выявленное отношение функции параллельного (последовательного) выбора реализует ее верхнюю (нижнюю) аппроксимацию.

Эти факты позволяют в ряде случаев решить задачи синтеза и аппроксимации, которые в общем случае нерешаемы (см. теорему 34).

ЛИТЕРАТУРА

1. *Murakami Y.* Logic and cocial choice. London: Routledge & Kegan Paul Ltd.; New York: Dover Publication Inc., 1968.
2. *Шоломов Л. А.* Логические методы в задачах согласованного выбора / Препринт ВНИИ системных исследований. М., 1978.
3. *Березовский Б. А., Барышников Ю. М., Борзенко В. И., Кемпнер Л. М.* Многокритериальная оптимизация: математические аспекты. М.: Наука, 1989.
4. *Макаров И. М., Виноградская Т. М., Рубчинский А. М., Соколов В. Б.* Теория выбора и принятия решений. М.: Наука, 1982.
5. *Левченков В. С.* Алгебраический подход в теории группового выбора. М.: Наука, 1990.
6. *Aleskerov F. T., Vladimirov A. V.* Hierarchical voting // Information sciences. 1986. V. 39. P. 41–86.
7. *Шоломов Л. А.* Логические методы исследования дискретных моделей выбора. М.: Наука, 1989.
8. *Айзерман М. А., Вольский В. И., Литваков Б. М.* Элементы теории выбора. Псевдокритерии и псевдокритериальный выбор. М.: Нефтяник, 1994.
9. *Вольский В. И., Лезина З. М.* Голосование в малых группах: процедуры и методы сравнительного анализа. М.: Наука, 1991.
10. *Шоломов Л. А.* Применение логических методов в задачах последовательного выбора / Препринт. ВНИИ системных исследований. М., 1980.
11. *Шоломов Л. А.* Логические методы композиции функций выбора / Препринт ВНИИ системных исследований. М., 1981.
12. *Шоломов Л. А.* Оценка сложностных характеристик одного механизма выбора с участием нескольких лиц // Изв. АН СССР. Техническая кибернетика. 1985. № 2. С. 3—13.
13. *Шоломов Л. А., Юдин Д. Б.* Сложность многошаговых схем обобщенного математического программирования // Изв. АН СССР. Техническая кибернетика. 1988. № 1. С. 13–22.

14. *Sholomov L.* Context-independent choice: description and analysis by means of first-order logic // Logic, Game theory and Social choice. Proceedings of the Intern. Conference LGS'99. Tilburg University Press, 1999. P. 549–559.

15. *Литваков Б. М.* Аппроксимация функций выбора // Автоматика и телемеханика. 1984. № 9. С. 138–146.

16. *Шоломов Л. А.* О сложности задач минимизации и сжатия моделей последовательного выбора // Дискретный анализ и исследование операций. Сер. 1. 1999. Т. 6. № 3. С. 87–109.

17. *Stockmeyer L. J.* The set-basis problem is NP-complete // Report N RC-5431. New York: IBM Research Center, Yorctown Heights, 1975.

18. *Шоломов Л. А.* О сложности реализации функций выбора системой отношений частичного порядка // Проблемы кибернетики. Вып. 41. М.: Наука, 1984. С. 111–116.

19. *Шоломов Л. А.* Функциональные возможности и сложность механизмов выбора, основанных на исключении худших вариантов // Изв. АН СССР. Техническая кибернетика. 1987. № 1. С. 10–17.

20. *Юдин Д. Б., Шоломов Л. А.* Многошаговые схемы обобщенного математического программирования и функции выбора // Докл. АН СССР. 1985. Т. 282. № 5. С. 1066–1069.

21. *Arrow K. J.* Difficulty in the concept of social welfare //J. Political Economy. 1950. V. 58. P. 326–346.

22. *Arrow K. J.* Social Choice and Individual Values, 2nd ed. New Haven—London: Yale University Press, 1963.

23. *Айзерман М. А., Алескеров Ф. Т.* Выбор вариантов: основы теории. М.: Наука, 1990.

24. *Фишберн П.* Теория полезности для принятия решений. М.: Наука, 1978.

25. *Шоломов Л. А.* Операторы над отношениями, сохраняющие транзитивность // Дискретная математика. 1998. Т. 10. Вып. 1. С. 28–45.

26. *Шоломов Л.А.* Исследование отношений в критериальных пространствах и синтез операторов группового выбора // Математические вопросы кибернетики. Вып. 5. М.: Физматлит, 1995. С. 109–143.

27. *Шоломов Л. А.* Агрегирование линейных порядков в задачах группового выбора // Автоматика и телемеханика. 1998. № 2. С. 113–122.

28. *Sholomov Lev A.* Explicit form of neutral social decision rules for basic rationality conditions // Mathematical Social Sciences. 2000. V. 39. No. 1. P. 81–107.

29. *Моркялюнас А.* Групповой выбор при независимости и слабой симметрии альтернатив // Математические методы в социальных науках. Вильнюс: Ин-т математики и кибернетики АН ЛитССР, 1985. Вып. 18. С. 57–60.

30. *Данилов В. И.* Модели группового выбора // Изв. АН СССР. Техническая кибернетика. 1983. № 1. С. 143–164.

31. *Владимиров А. В.* Исследование процедур построения коллективных решений: Автореф. дис. ... канд. техн. наук / Ин-т проблем управления. М., 1987.

32. *Шоломов Л. А.* О сложности реализации бинарных отношений путем теоретико-множественных операций над отношениями линейного порядка // Проблемы кибернетики. Вып. 41. М.: Наука, 1984. С. 101–109.

33. *Шоломов Л. А.* О представлении бинарного отношения набором критериев // Изв. АН СССР. Техническая кибернетика. 1984. № 1. С. 6–14.

34. *Hiraguchi T.* On the dimension of partially ordered sets // Sci Rep. Kanazawa University. 1951. V. 1. No. 2. P. 77–94.

35. *Оре О.* Теория графов. М.: Наука, 1980.

36. *Trotter W. T.* Embedding finite posets in cubes // Discrete Math. 1975. V. 12. No. 2. P. 165–172.

37. *Erdös P., Moser L.* On the representation of directed graphs as unions orderings // Magyar tud. akad. Mat. kutató int. közl. 1964. V. 9. No. 1–2. P. 125–132.

38. *Шоломов Л. А.* Декомпозиция отношений в задачах выбора: вполне разделимые отношения и независимость от пути // Автоматика и телемеханика. 2001. № 11. С. 154–164.

39. *Шоломов Л. А.* Анализ рациональности модели последовательного выбора // Автоматика и телемеханика. 2000. № 5. С. 124–132.

40. *Шоломов Л. А.* Представление и исследование порядковых моделей выбора средствами логики первого порядка // Математические вопросы кибернетики. Вып. 7. М.: Наука, Физматлит, 1998. С. 169–202.

41. *Мальцев А. И.* Алгебраические системы. М.: Наука, 1970.

42. *Булос Дж., Джеффри Р.* Вычислимость и логика. М.: Мир, 1994.

43. *Шоломов Л. А.* Сложность распознавания свойств порядковых отношений в n-мерных пространствах // Дискретный анализ и исследование операций. Сер. 1. 2002. Т. 9. № 4. С. 82–105.

44. *Подиновский В. В.* Многокритериальные задачи с однородными равноценными критериями // Журнал вычислительной математики и математической физики. 1975. Т. 15. № 2. С. 130–141.

45. *Подиновский В. В.* Многокритериальные задачи с упорядоченными по важности критериями // Автоматика и телемеханика. 1976. № 11. С. 118–127.

46. *Шоломов Л. А.* Синтез транзитивных порядковых отношений, согласованных с информацией о силе критериев // Сибирский журнал исследования операций. 1994. Т. 1. № 4. С. 64–92.

47. *Шоломов Л. А.* Метод интерпретаций в задаче синтеза операторов группового выбора // Алгебра и теория моделей 6. Новосибирск: НГТУ, 2007. С. 96–110.

48. *Шоломов Л. А.* Логические методы исследования отношений в критериальных пространствах с порядковыми шкалами произвольного вида // Автоматика и телемеханика. 2004. № 5. С. 120–130.

49. *Шоломов Л. А.* Распознавание свойств порядковых отношений в дискретных пространствах // Дискретный анализ и исследование операций. Сер. 1. 2004. Т. 11. № 3. С. 88–110.

50. *Шоломов Л. А.* Теоретико-модельный подход к описанию порядковых отношений в конечнозначных пространствах // Синтаксис и семантика логических систем: Материалы Российской школы-семинара, посвященной Ю. Е. Шишмареву. Владивосток: Дальнаука, 2008. С. 28–29.

ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ В ДИСКРЕТНОЙ МАТЕМАТИКЕ

DOI 10.17223/20710410/3/4
УДК 519.7

ПРИБЛИЖЁННОЕ РЕШЕНИЕ ЗАДАЧИ КОММИВОЯЖЕРА МЕТОДОМ РЕКУРСИВНОГО ПОСТРОЕНИЯ ВСПОМОГАТЕЛЬНОЙ КРИВОЙ

В. И. Дулькейт, Р. Т. Файзуллин

Омский государственный технический университет, г. Омск

E-mail: vidulkeyt@mail.ru

Предлагается эвристический алгоритм решения «задачи коммивояжёра», дающий приближённое решение.

Ключевые слова: *задача коммивояжёра, приближенное решение.*

Введение

Задача коммивояжера является важной и трудно решаемой [1]. Она возникает в обширном классе приложений, включая распознавание траекторий и образов, построение оптимальных схем движения и др. Наиболее распространённый частный случай (он же считается наиболее простейшим) — задача коммивояжера с евклидовой метрикой. В этой постановке её можно сформулировать следующим образом: пусть на плоскости задано множество точек (отметок) $A_k(k = 0, \ldots, n-1)$, требуется найти перестановку отметок B_i, такую, что достигает минимума сумма:

$$S = \sum_{k=0}^{n-1} d(B_k, B_{(k+1) \bmod n}),$$

где $d(A, B)$ — евклидово расстояние между точками A и B.

В терминах теории графов [2] задача может быть сформулирована так: имеется полный взвешенный граф, в котором вес каждого ребра равен евклидовому расстоянию между соответствующими вершинами. Задача состоит в нахождении гамильтонова цикла, для которого сумма весов его рёбер минимальна.

1. Обзор существующих решений

Задача коммивояжера является NP-полной [1], поэтому алгоритмы решения этой задачи делятся на две группы: точные и приближенные. Все точные алгоритмы фактически представляют собой оптимизированный полный перебор вариантов. В некоторых случаях эти алгоритмы достаточно быстро находят решения, но в общем случае приходится перебирать все $n!$ циклов.

Из приближенных алгоритмов наиболее известен алгоритм «иди к ближайшему соседу» (ИБ). Идея этого алгоритма проста: начиная обход с произвольной вершины, на каждом шаге выбирается ближайшая из не пройденных ещё вершин. Этот алгоритм легко формализуем, для него показана сходимость по вероятности к оптимальному решению с ростом числа точек. Кроме того, существует верхняя оценка отношения результата алгоритма ИБ к точному решению [1].

В работе [3] рассматривается приближённый алгоритм, основанный на построении достаточно гладкой кривой, аппроксимирующей путь, во вспомогательном пространстве высокой размерности.

Целью данной работы является разработка приближённого алгоритма решения поставленной задачи, основанного на рекурсивном построении гладкой вспомогательной кривой, аппроксимирующей путь.

2. Приближённый алгоритм решения задачи коммивояжера

Основная идея предлагаемого алгоритма заключается в том, что для нахождения требуемой перестановки строится гладкая вспомогательная кривая с быстро убывающими коэффициентами.

Воспользуемся подходом, описанным в работе [3]. Будем рассматривать функционал от $2n$ переменных:

$$J(x_0,\dots,x_{n-1},y_0,\dots,y_{n-1}) = \sum_{k=0}^{n-1} \sqrt{(x_{(k+1)\bmod n} - x_k)^2 + (y_{(k+1)\bmod n} - y_k)^2}.$$

Задача коммивояжера сводится к задаче минимизации значения этого функционала на фиксированных парах координат.

Приравняв к нулю частные производные, получаем однородную алгебраическую систему уравнений:

$$\begin{pmatrix} A & 0 \\ 0 & A \end{pmatrix} \begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \tag{1}$$

где A – ленточная матрица с элементами:

$$\begin{aligned} A_{k\ k} &= \frac{1}{\sqrt{(x_{(k-1)\bmod n} - x_k)^2 + (y_{(k-1)\bmod n} - y_k)^2}} + \\ &+ \frac{1}{\sqrt{(x_{(k+1)\bmod n} - x_k)^2 + (y_{(k+1)\bmod n} - y_k)^2}}, \\ A_{k\ k\pm1} &= \frac{-1}{\sqrt{(x_{(k\pm1)\bmod n} - x_k)^2 + (y_{(k\pm1)\bmod n} - y_k)^2}}. \end{aligned}$$

Выражение (1) есть аппроксимация оператора краевой задачи на неравномерной сетке:

$$\begin{cases} -\widetilde{X}''(t) = 0, \\ -\widetilde{Y}''(t) = 0, \end{cases} \quad \widetilde{X}(0) = \widetilde{X}(2\pi), \widetilde{Y}(0) = \widetilde{Y}(2\pi), \tag{2}$$

где $\widetilde{X}(t)$ и $\widetilde{Y}(t)$ суть некоторые гладкие периодические функции, которые в узлах сетки аппроксимации принимают значения координат отметок в порядке их следования по искомому пути.

Будем искать $\widetilde{X}(t)$ и $\widetilde{Y}(t)$ в виде

$$\begin{aligned} \widetilde{X}(t) &= \sum_{k=0}^{N} \alpha_k v_k(t), \\ \widetilde{Y}(t) &= \sum_{k=0}^{N} \beta_k v_k(t), \end{aligned} \tag{3}$$

где $v_k(t)$ образуют семейство ортонормированных собственных функций оператора левой части (2), которым соответствуют собственные числа λ_k.

Тогда условие (2) запишется в виде

$$\begin{cases} \sum\limits_{k=0}^{N} \alpha_k \lambda_k v_k(t) = 0, \\ \sum\limits_{k=0}^{N} \beta_k \lambda_k v_k(t) = 0. \end{cases} \tag{4}$$

Минимум левой части (4), не обязательно равный нулю, достигается на некотором наборе коэффициентов (α_k, β_k), который можно сопоставить с одним из возможных путей. При этом длина полученного пути существенно зависит от скорости убывания коэффициентов.

Итак, требуется найти функции $\widetilde{X}(t)$ и $\widetilde{Y}(t)$ в виде (3), такие, что

1) в узлах сетки аппроксимации они принимают значения координат отметок, причём порядок, в котором отметки будут пройдены, заранее не известен;
2) коэффициенты из (3) должны доставлять минимум (среди всех возможных наборов коэффициентов, обладающих первым свойством) левой части выражения (4).

Пусть $v_k(t) = e^{ikt}$, тогда $\lambda_k = 1/k^2$, и для нахождения $\widetilde{X}(t)$ и $\widetilde{Y}(t)$ можно использовать алгоритм быстрого преобразования Фурье (БПФ). При этом минимальная скорость убывания коэффициентов в левой части (4) будет порядка $1/k^2$.

На основе приведённых рассуждений можно построить рекурсивный алгоритм решения поставленной задачи.

Предположим, что на текущий момент вычислены k коэффициентов и известен некоторый порядок обхода отметок. На очередной итерации требуется скорректировать этот порядок, уточнить коэффициенты с учётом нового порядка обхода и вычислить $(k+1)$-е коэффициенты (3).

Назовём параметризованную кривую, координаты точек которой суть значения функций $\widetilde{X}(t)$ и $\widetilde{Y}(t)$, *вспомогательной кривой* (ВК). Опустим из отметок перпендикуляры к ВК и для каждой отметки выберем кратчайший перпендикуляр. Основания полученных перпендикуляров назовём *отметочными точками* (ОТ). Они однозначно сопоставляются отметкам, а значит, порядок их следования по ВК задаёт новый порядок обхода отметок. Этим обеспечивается выполнение первого условия.

Рассмотрим разности между координатами отметок и координатами ОТ. По ним с помощью метода линейной интерполяции можно построить функции $\Delta X(t)$ и $\Delta Y(t)$, которые характеризуют, насколько значения функций $\widetilde{X}(t)$ и $\widetilde{Y}(t)$ отличаются от координат отметок в узлах сетки аппроксимации. Выполним процедуру БПФ для функций $\Delta X(t)$ и $\Delta Y(t)$ и отбросим коэффициенты с номерами, превосходящими $k+1$. Тем самым будут получены уточнения для уже вычисленных коэффициентов и $(k+1)$-е коэффициенты.

Заметим, что, зная текущую ВК (т.е. часть младших коэффициентов), можно определить порядок следования отметок и построить соответствующий путь. Тем самым будут определены старшие коэффициенты (3). Чем ближе известный порядок отметок к оптимальному решению, тем меньший вклад в левую часть (4) будут привносить старшие коэффициенты. Следовательно, критерием успешности текущей ВК будет длина цикла, построенного по порядку следования ОТ. Этот параметр легко отсле-

живать на каждой итерации и запоминать порядок отметок, соответствующий минимальной длине цикла.

Основная проблема в предложенном алгоритме заключается в нахождении первоначального порядка отметок и нескольких младших коэффициентов. Можно предложить следующий эвристический метод решения этой проблемы: построить эллипс такой, что сумма квадратов расстояний от отметок до эллипса будет минимальной, и принять построенный эллипс за ВК. Тем самым будут определены первые два коэффициента из (3).

Учитывая, что в дальнейшем младшие коэффициенты будут уточняться, процесс построения эллипса можно упростить, если предположить, что одна из полуосей эллипса лежит на прямой, обладающей тем свойством, что сумма квадратов расстояний от отметок до этой прямой минимальна, а координаты центра эллипса являются средними арифметическими координат проекций отметок на эту прямую. Тогда задача построения эллипса сводится к построению указанной прямой и нахождению полуосей эллипса.

Приведём вкратце метод решения первой задачи [4]. Будем искать прямую в виде $ax + by + 1 = 0$. Устремим к минимуму сумму квадратов расстояний от отметок до искомой прямой:

$$S(a,b) = \frac{\sum\limits_{k=0}^{n-1}(ax_k + by_k + 1)^2}{a^2 + b^2} \to \min.$$

Введём следующие обозначения:

$$A = \sum_{k=0}^{n-1} x_k y_k, \quad X = \sum_{k=0}^{n-1} x_k, \quad Y = \sum_{k=0}^{n-1} y_k, \quad X_2 = \sum_{k=0}^{n-1} x_k^2, \quad Y_2 = \sum_{k=0}^{n-1} y_k^2.$$

Приравниваем к нулю частные производные:

$$\frac{\partial S}{\partial a} = \frac{2(b(b^2-a^2)A + ab^2(X_2 - Y_2) - 2abY + (b^2-a^2)X - an)}{a^2+b^2} = 0; \tag{5}$$

$$\frac{\partial S}{\partial b} = \frac{2(a(a^2-b^2)A + ba^2(Y_2 - X_2) - 2abX + (a^2-b^2)Y - bn)}{a^2+b^2} = 0. \tag{6}$$

Учитывая конечность a и b, из (5) и (6) получаем

$$a\frac{\partial S}{\partial a} + b\frac{\partial S}{\partial b} = -\frac{2(aX + bY + n)}{a^2+b^2} = 0.$$

Переносом начала системы координат можно добиться, чтобы $X \neq 0$, и тогда

$$a = -\frac{Y}{X}b - \frac{n}{X}. \tag{7}$$

Подставив (7) в (5), получаем уравнение для b:

$$\begin{aligned} 0 = &\left[A\left(1 - \left(\frac{Y}{X}\right)^2\right) - \frac{Y}{X}(X_2 - Y_2)\right] b^3 + \\ &+ \left[\frac{2Y^2}{X} + X\left(1 - \left(\frac{Y}{X}\right)^2\right) - \frac{2nYA}{X^2} - \frac{n(X_2 - Y_2)}{X}\right] b^2 + \\ &+ \left[\frac{nY}{X} - \frac{n^2 A}{X^2}\right] b. \end{aligned}$$

В общем случае это уравнение имеет три корня, один из которых есть $b = 0$. Окончательный выбор корня можно сделать, вычислив для каждой пары a и b значение $S(a, b)$.

Для нахождения полуосей эллипса p и q можно применить метод наискорейшего (градиентного) спуска [5]. Соответствующий итеративный процесс может быть таким:

$$\begin{aligned} p_{i+1} &= p_i - \alpha \frac{L(p_i + \Delta, q_i) - L(p_i, q_i)}{\Delta}, \\ q_{i+1} &= q_i - \alpha \frac{L(p_i, q_i + \Delta) - L(p_i, q_i)}{\Delta}, \end{aligned} \qquad i = 0, \ldots, M,$$

где p_0, q_0, α, Δ и M — некие параметры алгоритма, а $L(p, q)$ — сумма квадратов расстояний от отметок до эллипса с полуосями p и q.

Подводя итог сказанному выше, приведём алгоритм приближённого решения задачи коммивояжёра — Алгоритм 1.

Алгоритм 1. Приближённое решение задачи коммивояжёра

Вход: координаты отметок.
Выход: искомая перестановка отметок.

1: Построить начальную ВК на основе эллипса.
2: **Цикл**
3: Выполнить обратное БПФ для набора коэффициентов ВК, получив тем самым координаты точек ВК.
4: Найти ОТ для текущей ВК. В простейшем случае это можно сделать, последовательно просмотрев массив отметок и массив точек ВК, выбирая в качестве ОТ самую близкую к текущей отметке точку ВК.
5: Порядок следования отметочных точек задаёт некоторую перестановку отметок, которая определяет гамильтонов цикл. Вычислить длину этого цикла.
6: **Если** длина цикла окажется минимальной из всех предыдущих, **то**
7: Запомнить соответствующую перестановку.
8: По динамике изменения длины цикла можно определить, когда завершить итеративный процесс (например, при монотонном увеличении длины цикла на протяжении последних нескольких итераций).
9: Заполнить временные массивы (ВМ), число элементов в которых равно числу точек ВК. ВМ заполняются следующим образом: в отметочных точках берётся разность между координатами отметки и соответствующих ОТ, остальные элементы ВМ получаем, проведя процедуру линейной интерполяции. Фактически в ВМ хранятся значения функций $\Delta X(t)$ и $\Delta Y(t)$ в узлах некоторой сетки.
10: Применить прямое БПФ к ВМ, тем самым получим коэффициенты разложения функций $\Delta X(t)$ и $\Delta Y(t)$. Далее, коэффициенты с номерами больше I+2, где I – номер итерации и итерации нумеруются с единицы, положить равными нулю.
11: Сложить коэффициенты, полученные на предыдущем шаге, с коэффициентами ВК, получив новый набор коэффициентов ВК.
12: **Вернуть** перестановку, соответствующую циклу минимальной длины (данная перестановка была получена на шаге 7 одной из итераций).

На каждой итерации данный алгоритм отбрасывает на один коэффициент меньше, чем на предыдущей. За счёт такого приёма удаётся адаптивно менять «направление

изменения формы» ВК. Поясним данный факт на примере. Допустим, перед началом итеративного процесса имеется некоторый эллипс, построенный на первом шаге. Далее, если сразу вычислить все коэффициенты, то полученная кривая будет просто представлять собой эллипс, по контуру которого «пустили» некоторую функцию. Причём там, где эта функция положительна, соответствующие точки кривой «выдвигаются наружу» по нормали к эллипсу, а там, где функция отрицательна, — «задвигаются внутрь» по той же нормали. Таким образом, видно, что «направление изменения формы» ограничено нормалями к исходному эллипсу. В то же время если использовать описанную процедуру, то форма кривой будет существенно меняться, поскольку «направление изменения формы» зависит от ВК предыдущей итерации и постоянно уточняется.

3. Оценка эффективности и трудоёмкости

Для оценки сложности приведённого алгоритма предположим, что n — количество отметок, m — количество точек ВК, а k — количество новых коэффициентов, полученных на очередной итерации, т. е. на I-й итерации отбрасываются коэффициенты с номером больше чем $kI + 2$.

Тогда в худшем случае (когда длина каждого нового цикла оказывается меньше, чем длина предыдущего) будет сделано m/k итераций. Сложность каждой итерации можно оценить следующим образом:

$$2\mathrm{O}\left(n \log_2 m\right) + \mathrm{O}\left(nm\right) + \mathrm{O}\left(n\right) + \mathrm{O}\left(m\right),$$

где первое слагаемое есть оценка сложности прямого и обратного БПФ, второе — оценка сложности поиска отметочных точек, третье — трудоёмкость всех операций с отметками и отметочными точками (построение цикла, вычисление его длины и т. д.), четвёртое — трудоёмкость всех операций над точками и коэффициентами ВК (заполнение ВМ, отбрасывание коэффициентов). Если пренебречь трудоемкостью первого шага (он выполняется один раз) и отбросить наименее значимые величины, получим следующую оценку сложности всего алгоритма:

$$\mathrm{O}\left(\frac{mn \log_2 m}{k}\right) + \mathrm{O}\left(\frac{nm^2}{k}\right).$$

Вычислительные эксперименты показали, что для минимальной длины цикла значение m должно быть одной из ближайших к n степеней двойки. Поэтому с увеличением количества отметок второе слагаемое будет превалировать над первым. Следовательно, в Алгоритме 1 процедура поиска отметочных точек — наиболее «узкое» место в плане быстродействия.

Для ускорения работы алгоритма можно на каждой итерации отбрасывать на несколько коэффициентов меньше, чем на предыдущей, т. е. на I-й итерации отбрасывать коэффициенты с номером больше $kI+2$, где k — некоторый параметр алгоритма.

Стоит подчеркнуть, что Алгоритм 1 приближённый. Поэтому платой за относительный выигрыш в трудоёмкости (для точных алгоритмов трудоёмкость имеет порядок $O(n!)$) будет приближённый характер решения.

В таблице приведены следующие данные: точное решение [6], результаты работы Алгоритма 1 при $k = 1$, результаты работы алгоритма, описанного в работе [3], и результат алгоритма «иди в ближайший».

Сравнение результатов различных алгоритмов с известными точными решениями

Пример и число отметок	Точное решение	Результат предложенного алгоритма	Результат алгоритма из работы [3]	Результат алгоритма ИБ
Berlin52	7542	7993	8731	8980
A280	2586	2929	3129	3169
Bier127	118282	121680	138887	129397
Ch130	6110	6484	7088	7467
fl1577	22249	26288	30963	29091
eil101	629	679	734	832
kroA100	21282	22010	22498	26762
st70	675	697	746	795
pr76	108159	115613	119489	153380
kroC100	20749	21354	22314	25810
eil51	426	448	469	512
d657	48912	54756	64909	61261
Ch150	6528	6877	7746	8141
Lin105	14379	15279	16629	20213
Pr1002	259045	297194	373187	322323

ЛИТЕРАТУРА

1. *Гэри М, Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.
2. *Оре О.* Теория графов. М.: Наука, 1980.
3. *Файзулин Р. Т., Файзулин Р. Р.* Гладкие приближения в задаче коммивояжёра // Таврический вестник информатики и математики. 2004. №27.
4. *Скарборо Д.* Численные методы математического анализа. М.: ГТТИ, 1934.
5. *Бахвалов Н. С., Жидков Н. П., Кобельков Г. М.* Численные методы. М.: Лаборатория Базовых Знаний, 2002.
6. `www.iwr.uni-eidelberg.de/groups/comopt/software/TSPLIB95/tsp/`

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КРИПТОГРАФИИ

DOI 10.17223/20710410/3/5

УДК 681.322

ПОИСК УПРОЩЕННОЙ МОДЕЛИ ПРОТОКОЛОВ ИНФРАСТРУКТУРЫ ЦИФРОВОЙ ПОДПИСИ С ИСПОЛЬЗОВАНИЕМ ВЕРИФИКАТОРОВ МОДЕЛЕЙ

С. Е. Прокопьев

г. Москва

E-mail: prsz@bk.ru

Важнейшим способом повышения безопасности информационных систем, использующих криптографические методы защиты информации, является эффективная декомпозиция кода криптографических сервисов на программные модули. Под эффективной декомпозицией понимается: 1) изоляция программных модулей разной степени критичности (т. е. имеющих разную степень опасности последствий от ошибок или программных закладок) друг от друга; 2) применимость одних и тех же программных модулей без модификации (и, как следствие, повторной верификации) в различных криптографических сервисах; 3) получение компактных, детерминированных, криптографически полных упрощенных моделей программных модулей. Одним из возможных подходов к решению проблемы эффективной декомпозиции является предварительный анализ набора протоколов, реализующих криптографический сервис, в рамках т. н. *UC-моделей*. В настоящей статье рассмотрена проблема декомпозиции криптографического сервиса «Цифровая подпись на базе PKI» в рамках указанного подхода и исследована возможность частичной автоматизации процедуры поиска его упрощенной модели с использованием *верификатора моделей NuSMV*.

Ключевые слова: *UC-модели, верификаторы моделей, NuSMV, цифровая подпись, PKI.*

Введение

К программным реализациям криптографических алгоритмов предъявляются особые требования, поэтому прикладная и криптографическая части приложения обычно реализуются разными разработчиками. Для повышения безопасности приложений, использующих криптографические методы защиты информации (снижения ущерба от ошибок прикладного разработчика и трудоемкости анализа безопасности), криптографический код необходимо [1] размещать внутри т. н. *криптографического периметра (криптоядра)*, изолированного от прикладного кода и решающего, помимо реализации криптографических алгоритмов, задачу создания защищенного канала ввода ключевой информации. Известно, что слабые реализации криптографических протоколов не менее опасны, чем слабые реализации криптографических примитивов (функции шифрования, вычисления и проверки имитовставки и цифровой подписи и т. д.), поэтому в криптоядро обычно помещаются не только примитивы, но и целые криптографические сервисы. Под криптографическим сервисом здесь понимается реализация одного или нескольких криптографических протоколов, совместно решающих некоторую задачу, вместе с реализацией служебных функций. Например, сервис «Проверка цифровой подписи», кроме самого алгоритма верификации цифровой подписи, будет включать

реализацию целого семейства протоколов инфраструктуры открытых ключей (PKI), а также функции управления ключевой информацией и настройки правил политики безопасности.

Размещение кода криптографических протоколов внутри криптоядра, очевидно, приводит к сильному увеличению размера последнего. В то же время в криптографических протоколах существуют части, не являющиеся сильно критичными с точки зрения безопасности. Например, известна атака типа downgrade, когда злоумышленник понижает стойкость используемого участниками криптокомплекта (ciphersuite) в протоколе TLS [2]. Очевидно, что эта угроза менее опасна, чем угроза компрометации ключей, так как участники изначально были согласны на использование любых криптокомплектов из числа ими заявленных. С учетом этого, задачу накопления сообщений фазы handshake и вычисления значения хэш-функции от их конкатенации можно рассматривать как слабокритичную часть протокола и вынести из криптоядра. Таким образом, перед программной реализацией необходим анализ криптографических протоколов на предмет выявления в них частей разной степени критичности.

Другая проблема, возникающая при размещении кода протоколов в криптоядре, — это необходимость модульной декомпозиции сложных криптографических протоколов. Часто разные сложные протоколы используют в качестве вспомогательных одни и те же более простые. Необходимо провести декомпозицию кода разных криптографических серисов на модули так, чтобы одни и те же программные модули криптоядра могли без модификации (и, как следствие, повторной верификации) быть применены при конструировании различных криптографических сервисов. Для этого свойства безопасности протоколов должны быть проанализированы в как можно более нефиксированных условиях, в предельном случае — в условиях неизвестного окружения.

Далее, для того чтобы сформулировать свойства безопасности всей системы или сложного сервиса (обычно в виде списка предотвращаемых угроз), разработчикам нужна некоторая простая и адекватная модель, имитирующая сложные реальные алгоритмы сервисов, — их упрощенные модели (абстракции). Внешнее поведение абстракции должно быть неотличимым (в некоторой заданной степени) от внешнего поведения реального протокола (т. н. *принцип симуляции*). При этом абстракции должны быть криптографически полными (cryptographically sound): из отсутствия уязвимостей в модели информационной системы на базе абстракций криптографических сервисов должно следовать отсутствие уязвимостей в модели информационной системы на базе протоколов, реализующих эти сервисы.

С учетом приведенных выше рассуждений, из всех существующих подходов к анализу безопасности криптографических протоколов нам наиболее интересны модели, в которых изначально заложен модульный подход к анализу безопасности сложных составных криптографических протоколов с использованием криптографически полных абстракций.

Такие модели есть. Это модели *с определениями безопасности симуляцией в неизвестном окружении*, далее их будем называть *UC-моделями* (universally composable) (термин из [3]). Среди формально-логических — это модель Линкольна—Митчелла—Митчелла—Скедрова (ЛММС) [4, 5], среди теоретико-сложностных — Канетти [3, 6–8] и Пфитцманн—Шунтера—Вайднера (ПШВ) [9, 10]. Связь между этими моделями безопасности установлена в [4]; фактически, они отличаются способами задания модели вычислений.

1. Модели анализа безопасности криптографических протоколов с определениями безопасности симуляцией в неизвестном окружении

В UC-моделях используются две модели вычислений: реальная (*real*) и идеальная (*ideal*). В реальной модели каждый участник Pi использует для решения требуемой задачи программу $Prot(i)$, реализующую алгоритм протокола. Доставку сообщений между программами $Prot(k)$ осуществляет злоумышленник A, который может произвольным образом модифицировать их. В идеальной модели существует специальный объект — доверенная идеальная функциональность (*ideal functionality*, ИФ), к которой каждый участник имеет конфиденциальный и аутентифицированный доступ и которая решает для них эту задачу некоторым идеальным образом. Т. е. ИФ реализует идеальные спецификации задачи. Например, задачу создания защищенного соединения (протоколы TLS, IPSec, SSH) можно специфицировать следующей идеальной моделью: участники передают свои сообщения вместе с идентификаторами адресатов сообщений идеальной функциональности (назовем ее «Защищенное соединение»), которая просто передает их по назначению, используя свои каналы связи с участниками. Протокол считается безопасным, если для любого обладающего заданными вычислительными возможностями злоумышленника, пытающегося извлечь информацию из процесса взаимодействия сторон в реальной модели, существует злоумышленник, получающий эту же информацию в идеальной модели. В модели ПШВ это обеспечивается требованием вычислительной неразличимости выходных распределений злоумышленников реальной и идеальной моделей. В модели Канетти используется специальный участник — *окружение (environment)*, который пытается угадать, с каким из злоумышленников он взаимодействует. При этом окружение может произвольным образом взаимодействовать со злоумышленником и реализациями сервиса в обеих моделях (в реальной — с программами протокола, в идеальной — с ИФ). Схема определения безопасности в модели Канетти отображена на рис. 1.

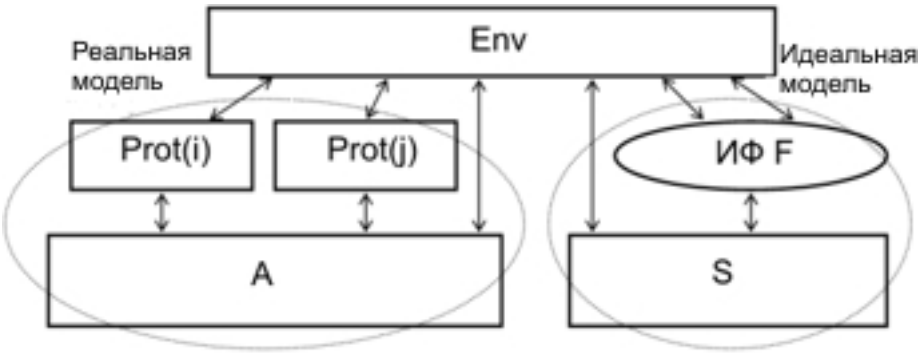


Рис. 1. Схема определения безопасности в модели Канетти

На схеме $Prot(i)$ и $Prot(j)$ — это программы, реализующие протокол на машинах участников i и j.

Немного более строгая формулировка определения безопасности в модели Канетти следующая:

Определение 1. *Протокол $Prot$ (совокупность $Prot(k)$ для всех участников $k = 1, 2, \ldots$) безопасно реализует ИФ F*, если для любого алгоритма A существует

алгоритм S, такой, что любое окружение Env не может отличить реальную модель от идеальной с вероятностью, существенно большей, чем $1/2$ [1].

Взаимодействие между ИФ F и злоумышленником идеальной модели S нужно для того, чтобы S мог получить информацию, которая с точки зрения безопасности решения задачи не является критической и получения которой злоумышленником A в реальных условиях не избежать. Например, в протоколах, реализующих задачу «Защищенное соединение», злоумышленник A всегда будет знать факт передачи, а также кому и кем направлены сообщения (так как он сам эти сообщения доставляет). Поэтому для обеспечения возможности симуляции необходимо, чтобы ИФ «Защищенное соединение» передавала злоумышленнику S эту информацию.

В дальнейшем при анализе сложных протоколов используется т. н. гибридная (*hybrid*) модель, в которой присутствуют как программы $Prot(k)$, так и ИФ. В гибридной модели программы $Prot(k)$ для решения вспомогательных задач обращаются к ИФ. При этом безопасность их композиции — сложносоставного протокола, полученного замещением ИФ протоколом, безопасно ее реализующим в «чистой» идеальной модели, — гарантируется теоремой о безопасности композиции безопасных протоколов.

Теорема 1 типовая для UC-подходов, неформальная формулировка. Если протокол $Prot1$ безопасно реализует ИФ $F1$ в $F2$-гибридной модели, а протокол $Prot2$ безопасно реализует ИФ $F2$ в реальной модели, то их композиция $Prot1(Prot2)$ безопасно реализует ИФ $F1$ в реальной модели.

Возможность такого результата обусловлена очень сильным определением безопасности — в неизвестном окружении, что приводит к сложности получения результатов в рамках UC-моделей. Поэтому необходимо развивать подходы, связанные с автоматизацией получения результатов.

2. Подготовка сервиса «Цифровая подпись на базе PKI» к автоматизированному анализу безопасности в рамках UC-моделей

Итак, UC-модели дают нам универсально применимую, компактную, детерминированную, криптографически полную абстракцию совокупности криптографических протоколов, решающих некоторую криптографическую задачу, — ИФ.

Определение безопасности в условиях неизвестного окружения и криптографическая полнота UC-моделей делают их наиболее мощными из всех существующих на сегодня подходов к анализу безопасности протоколов, т. е. покрывающими наибольшее число угроз безопасности. Обратной стороной этого является сложность получения результатов в рамках данного подхода, в частности автоматизации этого процесса. На вычислительной машине компактно и адекватно специфицировать такой объект как криптографический примитив, стойкий в смысле некоторого теоретико-сложностного определения безопасности, практически невозможно. В то же время одним из главных преимуществ UC-моделей является то, что формулировка ИФ криптографически полно симулирует поведение реального протокола, являясь при этом детерминированным алгоритмом. Поэтому, если примитивы будут заключены внутри вспомогательных под-

[1] Для большинства криптографических задач найдены протоколы, UC-безопасно их реализующие [3, 6, 7, 9]. Однако не для всех криптографических задач, для которых существует безопасный с точки зрения здравого смысла протокол, существует UC-безопасный протокол. Например, для нерешаемых в рамках UC-подхода задач привязки к биту, групповой подписи и некоторых других можно построить безопасные с точки зрения здравого смысла протоколы, безопасные в рамках т. н. игрового подхода к определению безопасности (эксперимента с оракулом).

протоколов нижнего уровня, то при анализе сложных протоколов эти подпротоколы можно заменить детерминированными ИФ без потери криптографической полноты (в отличие от алгебраических подходов к анализу безопасности криптографических протоколов на базе модели Долева—Яо). В итоге мы получаем детерминированный алгоритм сложного протокола, при анализе безопасности которого уже можно применять средства автоматизации.

В работе [11] в рамках модели ПШВ анализ безопасности протокола, реализующего задачу «Защищенное соединение с контролем порядка сообщений», был частично автоматизирован путем применения верификатора теорем PVS [12]. Данный протокол анализировался в гибридной модели, в которой в качестве вспомогательной ИФ выступала ИФ «Защищенное соединение без контроля порядка сообщений».

В настоящей работе в качестве объекта исследования было взято семейство протоколов инфраструктуры открытых ключей (PKI), реализующих сервис «Цифровая подпись на базе PKI». В качестве средства автоматизации был выбран верификатор моделей (*model checker*) NuSMV [13].

В [7] была сформулирована ИФ $F(Sig)$, привязывающая сообщения к открытому ключу. Наша цель — сформулировать надстройку над ней — функциональность, которая будет покрывать дополнительную специфику инфраструктуры PKI: привязку сообщений к идентификаторам субъектов, отзыв ключа, изменение действительности ключа во времени. Обозначим ее как ИФ $F(SigPKI)$. Вся вероятностная специфика реальных алгоритмов цифровой подписи скрыта внутри детерминированной $F(Sig)$, поэтому работа протоколов PKI в $F(Sig)$-гибридной модели будет детерминированной и может быть автоматизирована. Другая цель — оценить сложность протокола, для которого поиск формулировки ИФ может быть автоматизирован с помощью верификатора моделей NuSMV.

Одна из главных проблем при использовании средств формальной верификации — это обеспечение корректности спецификаций по отношению к анализируемой реальной системе. Одним из вариантов ее решения является написание спецификаций сначала на языке высокого уровня, а затем автоматическая трансляция его в спецификации на языке верификатора моделей. Как было показано в [14], такой подход неэффективен. Более эффективным способом повышения гарантий корректности спецификаций представляется декомпозиция спецификаций на независимые компоненты, взаимодействующие друг с другом посредством передачи сообщений. Такой подход, во-первых, существенно облегчает проверку корректности спецификаций за счет независимости модулей, во-вторых, позволяет использовать одни и те же модули в спецификациях разных систем. В [15] была предложена адаптация модели Канетти, которая задает эффективный способ декомпозиции криптографических приложений на программные модули с возможностью независимого анализа безопасности каждого из них. В настоящей работе мы внедрим этот подход в процесс описания моделей на языке спецификаций верификатора моделей NuSMV.

В соответствии с определением безопасности UC-моделей для любого алгоритма злоумышленника реальной модели A необходимо сконструировать такой алгоритм злоумышленника идеальной модели S, что окружение не сможет отличить A от S. Будем использовать распространенный подход на базе «черного ящика»: S запускает внутри себя копию A и эмулирует для него среду реальной модели.

Будем использовать следующий порядок действий:

1) Моделируется протокол (реальная модель).

2) Фиксируется предположительная формулировка ИФ и алгоритма симуляции злоумышленника S на базе «черного ящика A» (идеальная модель).

3) Используя возможности верификатора, перебираются алгоритмы «черного ящика A» и проверяется (с помощью предикатов верификатора) неразличимость моделей для окружения. Если верификатор обнаруживает контрпример, то либо идеальная модель (формулировки ИФ и алгоритма симуляции S), либо реальная модель (протокол) корректируются и симуляция проверяется заново. (Какая модель корректируется, зависит от того, какую задачу мы решаем: поиск формулировки ИФ для протокола (семейства протоколов) или построение протокола, безопасно реализующего заданную ИФ. В нашем случае — это поиск формулировки ИФ.)

4) Если модель не просчитывается (либо по времени, либо по памяти), то модель сокращается и просчитывается снова в соответствии с указанными выше шагами.

О полноте и корректности при сокращении модели

Формулировки функциональностей в UC-моделях предназначены для доказательства безопасности, т. е. доказывается только криптографическая полнота (cryptographic soundness) формулировки. И хотя авторы стремятся сделать их максимально корректными (см., например, трансформации ИФ «Цифровая подпись» ($F(Sig)$) от [3] к [6] в модели Канетти), корректность (*correctness*) формулировки не является конечной целью. Злоумышленнику идеальной модели позволено получить информации больше, чем злоумышленнику реальной модели, т.е. в обратную сторону симуляции нет. Поэтому, если в модели Канетти с точки зрения интуитивного представления о сервисе есть уязвимость в формулировке ИФ, это не означает, что она есть и в реальном протоколе, который ее безопасно реализует. Например, в реальных протоколах цифровой подписи вероятность того, что открытые ключи разных участников совпадут, пренебрежимо мала, в то время как в формулировке ИФ $F(Sig)$ из [3] открытый ключ генерируется злоумышленником идеальной модели; очевидно, что последний может сгенерировать один и тот же открытый ключ для разных участников с вероятностью 1.

При сокращении модели ситуация обратная: теряется криптографическая полнота. Однако смысл автоматизации построения функциональности заключается не в автоматическом получении доказательства симуляции, а в проверке алгоритма симуляции злоумышленника идеальной модели S. Поэтому при сокращении модели важно стремиться к сохранению корректности: найденные автоматизированным средством отклонения симуляции не должны быть ложными, т.е. должны присутствовать и в несокращенной модели. В дальнейшем они будут учтены при ручном построении функциональности. Цель — исследовать интересующий «срез» модели, повысить вероятность корректности доказательства, частично верифицировать утверждение о симуляции.

Таким образом, в общем случае реализация сокращенной модели на языке спецификаций верификатора может быть и не полной, и не корректной. Очевидно, что при сокращении модели нужно стараться, чтобы одновременно вероятность нахождения реальной уязвимости была высока, а ложной уязвимости — мала. Необходимо выбирать оптимальное (относительно вычислительных мощностей) соотношение между компактностью формулировки и степенью ее полноты и корректности.

Анализ критичности компонентов сервиса «Цифровая подпись на базе PKI»

Различные варианты политики безопасности (ПБ) фактически означают использование в реализации сервиса разных криптографических протоколов, поэтому ИФ,

соответствующие каждому варианту ПБ, будут отличаться. Таким образом, сначала необходимо зафиксировать возможные операции участников PKI.

Пусть участник, подписывающий сообщения, оперирует следующими командами: «сгенерировать новую пару», «выбрать текущий рабочий ключ подписи», «подписать сообщение», «обновить сертификат», «отозвать ключ».

Пусть участник, проверяющий подпись, оперирует командами: «загрузить список доверенных участников», «загрузить открытый ключ из сертификата», «выбрать текущий рабочий открытый ключ», «загрузить интервал доверия списка отозванных сертификатов (CRL)», «проверить подпись», «загрузить CRL».

Необходимо определить, какие команды следует отдать злоумышленнику. Напомним, что один из критериев эффективности декомпозиции — это изолирование частей кода приложения разной степени криптографической критичности друг от друга. Необходимо максимально сокращать объем критичного кода путем выделения из него некритичного кода. Поэтому, чем больше функций мы сможем безопасно отдать злоумышленнику, тем легче будет верифицировать критичный код. При этом нельзя «перестараться»: спецификации должны соответствовать нашему интуитивному представлению о задачах сервиса. Например, очевидно, что определение списка ключей участников, которым участник доверяет как удостоверяющим центрам, отдавать злоумышленнику нельзя. Отметим, что передача части функций злоумышленнику не означает, что в реальной информационной системе они в действительности передаются злоумышленнику, — еще раз подчеркнем, что это просто способ эффективной декомпозиции кода на модули разной степени критичности.

1. Правила ПБ вводятся с учетом криптографических аспектов реализации сервиса «цифровая подпись на базе PKI», однако все это носит приблизительный характер. Поэтому стойкость протоколов PKI должна сохраняться даже при «неправильном» задании правил ПБ. Таким образом, при анализе безопасности следующие функции ПБ можно отдать злоумышленнику:

1) определение интервала обновления CRL;

2) определение срока истечения сертификата;

3) определение интервала, в течение которого запросы считаются свежими.

2. В инфраструктуре PKI отрицательный результат верификации сообщения не обязательно означает, что участник X не подписывал сообщение M. Пользователи одновременно могут обладать несколькими действительными сертификатами, и верификация действительной подписи неправильно выбранным ключом даст ошибку верификации. Вообще, на задаваемый сервису «цифровая подпись на базе PKI» вопрос: «Подписывал ли участник X сообщение M?» — могут быть получены следующие ответы: 1) да; 2) искажено сообщение или подпись, либо неправильно выбран открытый ключ; 3) срок годности открытого ключа истек; 4) срок годности сертификата открытого ключа истек; 5) использованный при верификации ключ содержится в CRL; 6) срок действия сертификата ключа, которым подписан CRL, истек; 7) срок доверия к CRL истек; 8) срок действия сертификата удостоверяющего центра (УЦ) истек, и т. д. Тогда, например, при получении ответа «срок годности открытого ключа истек» пользователь сервиса может запустить процедуру получения и верификации более свежего сертификата. С точки зрения эффективности разработки приложений (принцип максимальной независимости компонентов) часть приложения, использующая сервис, не должна подвергаться изменениям при изменении (развитии) инфраструктуры PKI, поэтому управление ключами целесообразно заключить внутри сервиса. И тогда от-

вет на заданный вопрос будет либо «да», либо «не знаю». Учитывая это, следующие функции ПБ также отдадим злоумышленнику:

1) инициация генерации новой пары;
2) выбор ключа верификации;
3) обновление сертификата чужого открытого ключа;
4) обновление сертификата своего открытого ключа.

Итоговые формулировки компонентов UC-модели

1. В [6] была сформулирована ИФ «Цифровая подпись» — $F(Sig)$, привязывающая сообщение к открытому ключу. Специфика ЭЦП на базе PKI заключается в изменении действительности открытых ключей во времени. Эта специфика не связана с манипуляциями со значениями подписей, особенностями генерации открытого ключа, поэтому формулировку ИФ «Цифровая подпись» в нашей экспериментальной модели упростим до следующей:

ИФ «Цифровая подпись», $F(Sig)$:

Содержит поля $owner = Pi$, $fid = SIGN\_FID$, $inst = pk$, значение которых фиксируется при запуске, и массив m одноразово-записываемых переменных, в которых сохраняются сообщения.

Генерация подписи: После получения во внешнем входе от субъекта Pi сообщения (M_SIGN, msg) сохраняет сообщение msg в свободной переменной массива m и возвращает во внешний выход субъекту Pi сообщение $(M\_SIGN\_R\_OK)$.

Проверка подписи: После получения во внешнем входе субъекта Pj сообщения (M_VERIFY, msg): если сообщение msg было ранее сохранено, то возвращает во внешний выход субъекту Pj ответ $(M\_VERIFY\_R\_OK)$, иначе $(M\_VERIFY\_R\_FAIL)$.

---

2. Протокол, реализующий сервис «Цифровая подпись на базе PKI».

В представленной ниже формулировке аспекты политики безопасности, связанные с самими алгоритмами работы участников протокола, зафиксированы следующим образом:

— Владелец ключа не устанавливает срок годности своих ключей.

— Владелец не проверяет истечение срока годности своего сертификата.

— Обновление сертификата: Владелец подписывает запрос на обновление любого своего ключа любым своим ключом. Получив запрос, УЦ проверяет, что оба ключа не отозваны и их сертификаты действительны.

— УЦ подписывает все корректные запросы без POP (proof of possession, доказательство знания секретного ключа).

— Срок годности CRL определяет УЦ.

— Срок доверия к сообщениям о статусе ключей верифицирующий участник устанавливает сам.

Программы протокола PKI работают в $F(Sig)$-гибридной модели.

2.1. Владелец открытого ключа, **PKIo**:

Работает в $FSig$-гибридной модели. Содержит поля $owner = Pi$ (на практике присваивается при создании, здесь — фиксирован сразу), $fid = PKIO\_FID$, $inst$ (здесь будет только один экземпляр, поэтому поле не будет использоваться).

Генерация подписи. После получения во внешнем входе команды (M_SIGN,msg) передает в коммуникационный выход сообщение $(M\_GETMYPK)$. Ожидает в коммуникационном входе сообщение $(M\_GETMYPK\_R\_OK, pk)$, после чего передает $FSig[inst = pk]$ команду (M_SIGN, msg). После получения от $FSig[inst = pk]$ ответа (M_SIGN_OK) возвращает во внешний выход сообщение (M_SIGN_R_OK).

Выдача запроса на получение сертификата. После получения в коммуникационном входе команды $(M\_UPDMYCERT, pk1, pk2)$ передает $FSig[inst = pk2]$ сообщение $(M\_SIGN, msg = (CERTREQ\_ID, pk1, owner))$. Ожидает от $FSig[inst = pk2]$ ответ $(M\_SIGN\_R\_OK)$ и возвращает в коммуникационный выход сообщение $(M\_UPDMYCERT\_R\_OK, msg)$.

Выдача запроса на отзыв ключа. После получения в коммуникационном входе команды $(M\_RVKMYPK, pk)$ передает $FSig[inst = pk]$ сообщение $(M\_SIGN, msg = (RVKREQ\_ID, pk)$. Ожидает ответ $(M\_SIGN\_R\_OK)$ и возвращает в коммуникационный выход сообщение $(M\_RVKMYPK\_R\_OK, msg)$.

2.2. Субъект, проверяющий подписанное сообщение, $PKIv$:

Работает в $F(Sig)$-гибридной модели. Содержит переменные $owner = Pj$, $fid = PKIV\_FID$, $inst$, ca_pk — открытый ключ УЦ, которому Pj доверяет, массив записей $(pk, subj, notafter, rvkst, rvkstexpt)$, где pk — ключ субъекта $subj, notafter$ — срок годности ключа, $rvkst$ — состояние отзыва ключа ($REVOKED$ или OK), $rvkstexpt$ — срок годности значения состояния отзыва.

Обновление сертификата. После получения в коммуникационном входе команды $(M\_UPDCERT, msg = (CERT\_ID, pk, subj, notafter))$ передает $FSig[inst = ca\_pk]$ команду (M_VERIFY, msg). Если получен ответ $(M\_VERIFY\_R\_OK)$, то находит запись вида $(pk, subj, *, *)$ и обновляет поле $notafter$, после чего возвращает в коммуникационный выход сообщение $(M\_UPDCERT\_R\_OK)$. Если получен ответ $(M\_VERIFY\_R\_FAIL)$, то возвращает в коммуникационный выход сообщение $(M\_UPDCERT\_R\_FAIL)$.

Обновление состояния отзыва ключа. После получения в коммуникационном входе команды $(M\_UPDRVKST, msg = (RVKST\_ID, pk, rvkst, rvkstexpt))$ передает $FSig[inst = ca\_pk]$ сообщение (M_VERIFY, msg). Если от $FSig[inst = ca\_pk]$ получен ответ $(M\_VERIFY\_R\_FAIL)$, либо получен $(M\_VERIFY\_R\_OK)$, но ($curtime > rvkstexpt$ или $rvkst = OK$ и $pk.rvkst = REVOKED$), то возвращает $(M\_UPDRVKST\_R\_FAIL)$. Иначе обновляет поле $pk.rvkst := rvkst$ и возвращает в коммуникационный выход сообщение $(M\_UPDRVKST\_R\_OK)$.

Проверка подписи. После получения во внешнем входе команды $(M\_VERIFY, msg, subj)$ передает в коммуникационный выход сообщение $(M\_GETPK, subj)$. После получения в коммуникационном входе ответа (M_GETPK_R, pk) находит запись вида $(pk, subj, *, *)$, проверяет, что $curtime \leqslant notafter$, $rvkst \neq REVOKED$ и $curtime \leqslant rvkstexpt$. Если какое-либо из условий не выполнено, то возвращает во

внешний выход сообщение ($M\_VERIFY\_R\_FAIL$). Иначе передает $FSig[inst = = pk]$ сообщение (M_VERIFY, msg). Если от $FSig[inst = pk]$ получен ответ $(M\_VERIFY\_R\_OK)$, то возвращает во внешний выход $(M\_VERIFY\_R\_OK)$, если $(M\_VERIFY\_R\_FAIL)$, то $(M\_VERIFY\_R\_FAIL)$.

2.3. Удостоверяющий центр, **PKIca**:

Работает в $FSig$-гибридной модели. Содержит поля $cert\_dur$ — длительность действия сертификатов, $rvkst\_dur$ — длительность действия сообщений о состоянии отзыва, $mypk$ — свой открытый ключ, массив записей $(pk, subj, notafter, rvkst)$, где $rvkst$ принимает значения OK и $REVOKED$.

Выдача сертификата. После получения в коммуникационном входе сообщения $(M\_CERTREQ, msg = (CERTREQ\_ID, pk1, subj), pk2)$ проверяет, что $pk1.subj = = subj, pk1.rvkst = OK$, $pk2.subj = subj$, $curtime \leqslant pk2.notafter$, $pk2.rvkst = OK$. Если какое-либо из условий не выполнено, то возвращает в коммуникационный выход сообщение $(M\_CERTREQ\_R\_FAIL)$. Иначе передает $FSig[inst = pk2]$ сообщение (M_VERIFY, msg). Если от $FSig[inst = pk2]$ получен ответ $(M\_VERIFY\_R\_OK)$, то обновляет поле $pk1.notafter := curtime + cert\_dur$ и передает $FSig[inst = mypk]$ сообщение $(M\_SIGN, msg' = (CERT\_ID, pk1, pk1.subj, curtime + cert\_dur))$. После получения ответа $(SIGN\_R\_OK)$ возвращает в коммуникационный выход ответ $(M\_CERTREQ\_R\_OK, msg')$.

Отзыв ключа. После получения в коммуникационном входе сообщения $(M\_RVKREQ, msg = (RVKREQ\_ID, pk))$ проверяет, что есть запись вида $(pk, *, *, rvkst)$, где $rvkst \neq REVOKED$. Если такой записи нет, то возвращает в коммуникационный выход сообщение $(M\_RVKREQ\_R\_FAIL)$. Иначе передает $FSig[inst = pk]$ сообщение (M_VERIFY, msg). Если получен ответ $(M\_VERIFY\_R\_FAIL)$, то возвращает в коммуникационный выход $(M\_RVKREQ\_R\_FAIL)$. Иначе присваивает $pk.rvkst := REVOKED$ и возвращает в коммуникационный выход сообщение $(M\_RVKREQ\_R\_OK, msg')$.

Выдача статуса отзыва. После получения в коммуникационном входе сообщения $(M\_RVKSTREQ, pk)$ передает $FSig[inst = mypk]$ сообщение $(M\_SIGN, msg' = = (RVKST\_ID, pk, rvkst, curtime + rvkst\_dur))$. После получения ответа $(M\_SIGN\_R\_OK)$ возвращает в коммуникационный выход сообщение $(M\_RVKSTREQ\_R\_OK, msg')$.

П р и м е ч а н и е: При выдаче сертификата УЦ PKIca проверяет, что ключ в запросе на сертификат ранее не был подписан другим субъектом. Это замена механизма проверки обладания секретным ключом (POP) для случая, когда УЦ в системе один. Если вводить POP, то это нужно делать путем модификации ИФ $FSig$: к команде (Verify) добавится команда (Get_POP).

3. Формулировка абстракции совокупности протоколов PKI — ИФ «Цифровая подпись на базе PKI», $F(SigPKI)$:

Содержит поля $owner = Pi$, $fid = SIGNPKI\_FID$, значение которых фиксируется при запуске, и массив одноразово-записываемых записей вида (m, pk).

Генерация подписи. После получения во внешнем входе от участника Pi сообщения (M_SIGN, m) записывает в коммуникационный выход сообщение $(M\_GETMYPK)$.

После получения ответа $(M\_GETMYPK\_R\_OK, pk)$ сохраняет запись (m, pk) и возвращает во внешний выход сообщение $(M\_SIGN\_R\_OK)$.

Генерация подписи сообщений PKI. После получения в коммуникационном входе сообщения (M_SIGN, m, pk) проверяет, что первое поле m равно либо $CERTREQ\_ID$, либо $RVKREQ\_ID$. Если нет, то возвращает в коммуникационный выход сообщение $(M\_SIGN\_R\_FAIL)$. Иначе сохраняет запись (m, pk) и возвращает в коммуникационный выход ответ $(M\_SIGN\_R\_OK)$.

Проверка подписи. После получения во внешнем входе сообщения (M_VERIFY, m) записывает в коммуникационный выход сообщение (M_GETPK). Если получен ответ $(M\_GETPK\_R\_FAIL)$, то возвращает во внешний выход ответ $(M\_VERIFY\_R\_FAIL)$. Если получен ответ $(M\_GETPK\_R\_OK, pk)$, то находит сохраненную запись (m, pk) и возвращает во внешний выход (M_VERIFY_OK). Если такой записи нет, то возвращает $(M\_VERIFY\_R\_FAIL)$.

Проверка подписи злоумышленником. После получения в коммуникационном входе сообщения (M_VERIFY, m, pk) находит сохраненную запись (m, pk) и возвращает в коммуникационный выход (M_VERIFY_OK). Если такой записи нет, то возвращает $(M\_VERIFY\_R\_FAIL)$.

Отметим, что можно было не включать в $FSigPKI$ для каждой сохраненной строки текущий открытый ключ, а возложить задачу запоминания того, какой ключ соответствует каждому сообщению на злоумышленника, и тогда формулировка ИФ была бы более компактной. Однако в этом случае значительно возрастет сложность легального алгоритма работы злоумышленника, без которого протокол не будет работать. Наша главная цель — свертывание большого протокола в компактную ИФ, однако легальный алгоритм работы злоумышленника, по возможности, тоже должен быть компактным.

4. Пусть A' — «черный ящик», реализующий алгоритм работы злоумышленника идеальной модели. Зададим следующий алгоритм симуляции злоумышленника идеальной модели, S:

При инициализации S имеет значения массивов переменных $pv0.(pk, expt, rvkst)$, $pca0.(pk, subj, expt)$, переменных $po0.owner$,$pca0.cert\_dur$, $pca0.rvkst\_dur$.

A' посылает $PKIo$ $(M\_UPDMYCERT, pk1, pk2)$: S посылает $FSigPKI$ сообщение $(M\_SIGN, msg = (CERTREQ\_ID, pk1, po0.owner), pk2)$, ожидает ответ $(M\_SIGN\_R\_OK)$, возвращает A' сообщение $(M\_UPDMYPK\_R\_OK, msg)$.

A' посылает $PKIca(M\_CERTREQ, msg = (CERTREQ\_ID, pk1, subj), pk2)$: Если $pca0.pk1.subj \neq subj$, или $curtime > pca0.pk2.expt$, или $curtime \geqslant pca0.pk2.rvkst$, или $pca0.pk2.subj \neq subj$, то возвращает A' $(M\_CERTREQ\_R\_FAIL)$. Иначе посылает $FSigPKI$ сообщение $(M\_VERIFY, msg, pk2)$. Если от $FSigPKI$ получен ответ $(M\_VERIFY\_R\_OK)$, то возвращает A' сообщение $(M\_CERTREQ\_R\_OK, msg' = (CERT\_ID, pk1, subj, curtime + pca0.cert\_dur))$ и запоминает запись $(pk1, subj, curtime + pca0.cert\_dur)$. Если получен ответ $(M\_VERIFY\_R\_FAIL)$, то возвращает A' сообщение $(M\_CERTREQ\_R\_FAIL)$.

A' посылает $PKIca$ $(M\_RVKREQ, msg = (RVKREQ\_ID, pk))$: S посылает $FSigPKI$ сообщение (M_VERIFY, msg, pk). Если от $FSigPKI$ получен ответ $(M\_VERIFY\_R\_OK)$, то присваивает $pca0.pk.rvkst := REVOKED$ и воз-

вращает A' сообщение ($M\_RVKREQ\_R\_OK$). Иначе возвращает A' сообщение ($M\_RVKREQ\_R\_FAIL$).

A' посылает $PKIca$ ($M\_RVKSTREQ, pk$): S возвращает A' сообщение ($M\_RVKREQ\_R\_OK$, $RVKST\_ID$, $msg' = (pk, pca0.pk.rvkst, curtime + pca0.rvkst\_dur)$), запоминает msg' и возвращает A' сообщение ($M\_RVKSTREQ\_R\_OK, msg'$).

A' посылает $PKIv$ ($M\_UPDCERT, msg = (CERT\_ID, pk, subj, expt)$): Если у S запись $(pk, subj, expt)$ не была сохранена, либо $pv0.pk.subj \neq subj$, то возвращает A' ($M\_UPDCERT\_R\_FAIL$). Иначе обновляет значение $pv0.pk.expt := expt$ и возвращает A' ($M\_UPDCERT\_OK$).

A' посылает $PKIv$ ($M\_UPDRVKST$, $msg = (RVKST\_ID, pk, rvkst, rvkstexpt)$): Если у S запись $(pk, rvkst, rvkstexpt)$ не была сохранена, либо $pv0.pk.rvkst = OK$ и $rvkst = REVOKED$, либо $curtime > rvkstexpt$, то возвращает A' ($M\_UPDRVKST\_R\_FAIL$). Иначе обновляет значение $PKIv.pk.rvkst := rvkst$ и возвращает A' ($M\_UPDCERT\_OK$).

$FSigPKI$ посылает S (M_GETPK): S посылает A' (M_GETPK), ожидает ответ ($M\_GETPK\_R\_OK, pk$). Если $curtime \leqslant pk.expt$ и $pk.rvkst = OK$, то возвращает $FSigPKI$ ответ ($M\_GETPK\_R\_OK, pk$), иначе ($M\_GETPK\_R\_FAIL$).

$FSigPKI$ посылает S ($M\_GETMYPK$): S посылает A' ($M\_GETMYPK$), ожидает ответ ($M\_GETMYPK\_R\_OK, pk$) и возвращает его $FSigPKI$.

3. Эксперименты с верификатором моделей NuSMV

Сокращение мощности модели

Приведенные выше формулировки алгоритмов участников протокола, алгоритмов идеальных функциональностей и алгоритма симуляции злоумышленником не ограничены в количестве хранимых внутри себя сообщений. Для возможности задания спецификаций модели на верификаторе моделей необходимо привести модель в конечный вид. Для сокращения мощности модели были использованы следующие приемы:

1. Число копий $FSig$ — 4. Массив сообщений, хранимых каждой копией $FSig$, содержит три элемента. Вычисление модели останавливается, когда производится попытка записи сообщения в уже заполненную копию ИФ $FSig$.

2. Число копий $PKIo$ — 1. $PKIo$ содержит два собственных ключа.

3. Число копий $PKIca$ — 1. $PKIca$ имеет один собственный ключ.

4. Число копий $PKIv$ — 1. В $PKIv$ фиксирован открытый ключ УЦ.

5. Для сокращения вариантов выбора использовались следующие алгоритмы генерации сообщений злоумышленником реальной модели A (в нотации NuSMV):

alg1 (полный перебор, моделирование недетерминированного алгоритма работы A): $init(m[0]) := \{0,1\}$; $init(m[1]) := \{0,1,2,3\}$; $init(m[2]) := \{0,1\}$; $init(m[3]) := \{0,1,2,3\}$; $init(m[4]) := \{0,1\}$; $next(m[0]) := \{0,1\}$; $next(m[1]) := \{0,1,2,3\}$; $next(m[2]) := \{0,1\}$; $next(m[3]) := \{0,1,2,3\}$; $next(m[4]) := \{0,1\}$;

alg2: $init(m[0]) := 0$; $init(m[1]) := 0$; $init(m[2]) := 0$; $init(m[3]) := 0$; $init(m[4]) := 0$; $next(m[0]) := m[2]$; $next(m[1]) := \{0,1,2,3\})$; $next(m[2]) := \{0,1\}$; $next(m[3]) := m[1]$; $next(m[4]) := \{0,1\}$;

alg3: $init(m[0]) := 1$; $init(m[1]) := 0$; $init(m[2]) := 1$; $init(m[3]) := 0$; $init(m[4]) := 0$; $next(m[0]) := m[2]$; $next(m[1]) := \{0,1,2,3\}$; $next(m[2]) := \{0,1\}$; $next(m[3]) := m[1]$; $next(m[4]) := \{0,1\}$;

alg4: $init(m[0]) := \{0,1\}$; $init(m[1]) := \{0,1,2,3\}$; $init(m[2]) := \{0,1\}$; $init(m[3]) := \{0,1,2,3\}$; $init(m[4]) := \{0,1\}$; $next(m[0]) := m[2]$; $next(m[1]) := m[0] + m[2]$; $next(m[2]) := \{0,1\}$; $next(m[3]) := m[1]$; $next(m[4]) := \{0,1\}$;

alg5: $init(m[0]) := \{0,1\}$; $init(m[1]) := \{0,1,2,3\}$; $init(m[2]) := \{0,1\}$; $init(m[3]) := \{0,1,2,3\}$; $init(m[4]) := \{0,1\}$; $next(m[0]) := a.m[2]$; $next(a.m[1]) := a.m[3]$; $next(a.m[2]) := a.m[0]$; $next(a.m[4]) := \{0,1\}$;

6. Использовался следующий алгоритм изменения переменной текущего времени $curtime$:

$next(curtime) := case$

$(token = CHECK\_SIM\_E | token = CHECK\_SIM\_A) \& curtime = 0 : 1;$

$(token = CHECK\_SIM\_E | token = CHECK\_SIM\_A) \& curtime = 1 : 2;$

$1 : curtime; esac;$

Результаты экспериментов

В результате просчитывания модели на верификаторе моделей NuSMV (в указанных выше вариантах сокращения мощности) нарушений симуляции, обеспечиваемой предложенным алгоритмом работы злоумышленника A, выявлено не было. Ниже в таблице представлена зависимость времени вычисления и объема модели в памяти от алгоритмов работы злоумышленника реальной модели A (для AMD Sempron 2800+, 512 Mb RAM, Linux 2.4.18):

№	A	NumAll	NumReach	MemGo	MemCheck	TimeGo	TimeCheck
1	alg1	$2^{265.343}$	*	*	*	*	*
2	alg2	$2^{265.343}$	348982	196	410	25:22	45:32
3	alg3	$2^{265.343}$	180033	157	380	12:28	23:28
4	alg4	$2^{265.343}$	15487	56	90	1:14	1:41
5	alg5	$2^{265.343}$	9708	53	81	0:53	1:15

Здесь A — алгоритм работы злоумышленника реальной модели, NumAll — число всех состояний модели, NumReach — число достижимых состояний модели, MemGo — размер модели в памяти (фаза *go*, в мегабайтах), MemCheck — размер модели в памяти(фаза $compute\_reachable + check\_spec$, в мегабайтах), TimeGo — время вычисления (фаза *go*, мин:с), TimeCheck — время вычисления (фаза $compute\_reachable + check\_spec$, мин:с), * — исчерпан объем оперативной памяти через 1 час.

Выводы

1. Был сформулирован вариант компактной детерминированной модели (идеальной функциональности) семейства криптографических протоколов, реализующих сервис «Цифровая подпись на базе PKI», а также алгоритм симуляции злоумышленника идеальной модели, необходимый для проверки корректности данной формулировки в рамках UC-моделей.

2. Эксперименты с верификатором моделей NuSMV позволили оценить сложность протокола, для которого поиск формулировки ИФ может быть автоматизирован с помощью современных верификаторов моделей.

3. По результатам экспериментов, сокращение модели вычислений для верификаторов моделей рекомендуется проводить в 2 этапа. На первом этапе из реальной модели убираются все функции, которые, на взгляд исследователя, не связаны с интересующими его свойствами полной модели. Полученная сокращенная модель описывается на языке спецификаций верификатора. Цель сокращения модели на первом этапе

— уместить ее символическое представление в памяти (не должно занимать больше, чем 1/2 объема оперативной памяти — см. таблицу результатов экспериментов). На втором этапе вводятся искусственные ограничения на перебираемые верификатором состояния модели. Например, устраняется логическое дублирование ветвей вычислений, когда изменение порядка некоторых действий в системе на результат анализа не влияет. Если в результате работы на втором этапе модель становится быстропросчитываемой, тогда в модель возвращаются некоторые функции, сокращенные на первом этапе, после чего снова проводится второй этап, и т. д.

ЛИТЕРАТУРА

1. *NSA Cross-Organizational CAPI Team.* Security Service API: Cryptographic API Recommendation. Second Edition. // Government Information Technology Issues, 1996.
2. RFC 2246. The TLS Protocol Version 1.0. January 1999.
3. *Canetti R.* Universally Composable Security: A New paradigm for Cryptographic Protocols // 42nd FOCS, 2001.
4. *Lincoln P., Mitchell J., Mitchell M., Scedrov A.* Probabilistic Polynomial-Time Equivalence and Security Analysis. `http://citeseer.ist.psu.edu.` 1999.
5. *Mateus P., Mitchell J., Scedrov A.* Composition of Cryptographic Protocols in a Probabilistic Polynomial-Time Process Calculus. `http://citeseer.ist.psu.edu.` 2002.
6. *Canetti R.* Universally Composable Signature, Certification, and Authentication. `http://citeseer.ist.psu.edu.` 2004.
7. *Canetti R., Krawczyk H.* Universally Composable Notions of Key Exchange and Secure Channels. `http://citeseer.ist.psu.edu.` 2002.
8. *Canetti R., Rabin T.* Universal Composition with Join State. `http://eprint.iacr.org.` 2002.
9. *BackesM., Pfitzmann B., Waidner M.* A Universally Composable Cryptographic Library. `http://citeseer.ist.psu.edu.` 2003.
10. *Pfitzmann B., Schunter M., Waidner M.* Secure Reactive Systems. // IBM Research Report RZ 3206(93252), IBM Research Division, Zurich, Feb. 2000.
11. *Backes M., Jacobi C.* Cryptographically Sound and Machine-Assisted Verification of Security Protocols // In Proc. 20th Annual STACS, volume 2607 of Lecture Notes in Computer Science, pages 675-686. Springer, 2003.
12. `http://pvs.csl.sri.com` — Prototype Verification System (PVS).
13. *Cimatti A., Clarke E., Giunchiglia F., Roveri M.* NuSMV: a new symbolic model checker. // Software Tools for Technology Transfer, 1998.
14. *Cheetancheri S.* Our experiments with NuSMV. `http://shasta.cs.ucdavis.edu/eas.` 2004.
15. *Прокопьев С. Е.* Адаптация модели безопасности Канетти для использования в качестве архитектуры подсистемы криптографических сервисов // Проблемы информационной безопасности. Компьютерные системы. 2005. № 5.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

DOI 10.17223/20710410/3/6

УДК 004.94

ПОДХОДЫ К ПОСТРОЕНИЮ ДП-МОДЕЛИ ФАЙЛОВЫХ СИСТЕМ

П. В. Буренин

ООО «Твест», г. Тверь

E-mail: troy1f4@mail.ru

В статье приводятся подходы к созданию ДП-модели файловых систем, в основе которой используется семейство ДП-моделей компьютерных систем с дискреционным управлением доступом. В рамках ДП-модели рассматриваются специфичные для файловых систем условия функционирования субъектов, условия передачи прав доступа и реализации информационных потоков, а также обосновываются достаточные условия реализации в файловых системах запрещенных информационных потоков по памяти.

Ключевые слова: *компьютерная безопасность, файловые системы, ДП-модель.*

С целью обеспечения возможности анализа условий получения недоверенными субъектами контроля над доверенными субъектами, реализующими механизмы защиты файловых систем (ФС), или условий создания недоверенными субъектами информационных потоков по памяти в обход механизмов защиты ФС построим на основе ДП-модели с функционально-ассоциированными с субъектами сущностями (ФАС ДП-модели, [1]) и разработанной Д. Н. Колеговым модели с функционально- и параметрически-ассоциированными с субъектами сущностями с дискреционным управлением доступом (ФПАС ДП-модели) ДП-модель файловых систем (или, сокращенно, ФС ДП-модель).

При этом для построения ФС ДП-модели в ФАС и ФПАС ДП-модели внесены изменения, позволяющие учитывать существенные особенности реализации механизмов защиты современных ФС. Таким образом, в дальнейшем используем следующее предположение.

Предположение 1. В рамках ФС ДП-модели выполняются следующие условия.

Условие 1. Во множестве сущностей выделено подмножество сущностей, защищенных ФС и не являющихся субъектами.

Условие 2. Во множестве доверенных субъектов выделено подмножество субъектов, обладающих правами доступа и реализующих доступ к сущностям, защищенным ФС, и кодирование в них данных в случае, когда оно осуществляется ФС. Эти доверенные субъекты реализуют информационные потоки по памяти между каждой сущностью, защищенной ФС, и соответствующей ей сущностью-образом, не являющейся субъектом.

Условие 3. Доверенные или недоверенные субъекты, не реализующие доступ к сущностям, защищенным ФС, не обладают правами доступа и не могут получать доступ к этим сущностям. При этом они могут обладать правами доступа или получать доступ к сущностям-образам сущностей, защищенных ФС.

Условие 4. В каждом состоянии системы кроме множества субъектов анализируется множество потенциальных доверенных субъектов (доверенных субъектов, которые могут быть созданы в процессе функционирования системы для реализации доступа к сущностям, защищенным ФС).

Условие 5. Кроме возможности создания новых субъектов из сущностей недоверенный субъект может создать доверенного субъекта в случае, когда недоверенный субъект реализовал к себе информационные потоки по памяти от всех сущностей, параметрически ассоциированных с потенциальным доверенным субъектом. При этом недоверенный субъект получает контроль над созданным доверенным субъектом.

Условие 6. Каждый доверенный субъект не обладает правами доступа ко всем сущностям.

Условие 7. Доверенные субъекты, не реализующие доступ к сущностям, защищенным ФС, в процессе функционирования системы не получают новые доступы к сущностям и не участвуют в создание информационных потоков к или от сущностей, защищенных ФС.

Условие 8. Не рассматриваются информационные потоки по времени, право доступа и доступ на запись в конец сущности.

Условие 9. В начальном состоянии системы недоверенные субъекты не реализуют доступы к сущностям, к ним не имеют доступы другие субъекты и отсутствуют информационные потоки по памяти с участием недоверенных субъектов.

В основе ФС ДП-модели использован классический подход (используемый, в том числе, в семействе ДП-моделей КС с дискреционным, мандатным или ролевым управлением доступом), состоящий в том, что каждая моделируемая КС представляется абстрактной системой, каждое состояние которой представляется графом доступов, каждый переход системы из состояния в состояние осуществляется в результате применения одного из правил преобразования графов доступа.

В рамках предположения 1 используем следующие обозначения и определения ФАС и ФПАС ДП-моделей:

— $E = O \cup C$ — множество сущностей, где O — множество объектов, C — множество контейнеров и $O \cap C = \varnothing$;

— $S \subset E$ — множество субъектов;

— $[s] \subset E$ — множество всех сущностей, функционально ассоциированных с субъектом s (при этом по определению выполняется условие $s \in [s]$, и для каждого субъекта множество сущностей, функционально с ним ассоциированных, не изменяется в процессе функционирования системы);

— $]s[\subset E$ — множество всех сущностей, параметрически ассоциированных с субъектом и потенциальным субъектом s (при этом по определению для каждого субъекта множество сущностей, параметрически с ним ассоциированных, не изменяется в процессе функционирования системы);

— L_S — множество доверенных субъектов;

— N_S — множество недоверенных субъектов, при этом по определению выполняется равенство $L_S \cap N_S = \varnothing$;

— $R_r = \{read_r,\ write_r,\ execute_r,\ own_r\}$ — множество видов прав доступа;

— $R_a = \{read_a,\ write_a\}$ — множество видов доступа;

— $R_f = \{write_m\}$ — множество видов информационных потоков, где $write_m$ — информационный поток по памяти на запись в сущность.

Определение 1. Иерархией сущностей называется заданное на множестве сущностей E отношение частичного порядка «$\leqslant$», удовлетворяющее условию:

если для сущности $e \in E$ существуют сущности e_1, $e_2 \in E$, такие, что $e \leqslant e_2$, $e \leqslant e_1$, то $e_1 \leqslant e_2$ или $e_2 \leqslant e_1$.

В случае, когда для двух сущностей e_1, $e_2 \in E$ выполняются условия $e_1 \leqslant e_2$ 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 $H : E \to 2^E$ — функцию иерархии сущностей, сопоставляющую каждой сущности $c \in E$ множество сущностей $H(c) \subset E$ и удовлетворяющую следующим условиям:

Условие 1. Если сущность $e \in H(c)$, то $e < c$ и не существует 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 $H(e_1) \cap H(e_2) = \varnothing$ и условия:

— если $o \in O$, то выполняется равенство $H(o) = \varnothing$;
— 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 $H(e) \subset E \setminus S$;
— если $s \in S$, то $H(s) \subset S$.

В рамках предположения 1 в ФС ДП-модели дополнительно используем следующие обозначения:

— $FSE \subset E \setminus S$ — множество сущностей, защищенных ФС;

— fs: $FSE \to E \setminus S$ — инъективная функция, которая ставит в соответствие каждой сущности, защищенной ФС, соответствующую ей сущность-образ;

— PS — множество потенциальных доверенных субъектов, реализующих доступ к сущностям из множества FSE;

— $FSS \subseteq L_S \cap S$ — множество доверенных субъектов, реализующих доступ к сущностям из множества FSE.

Будем считать, что в дальнейшем выполняется следующее предположение.

Предположение 2. В рамках ФС ДП-модели выполняются следующие условия.

Условие 1. Каждый доверенный субъект из множества FSS является функционально корректным, корректным относительно любой сущности и может обладать только правами доступа на чтение и запись к сущностям из множества FSE и соответствующим им сущностям-образам.

Условие 2. Каждый потенциальный доверенный субъект из множества PS может обладать только правами доступа на чтение и запись к сущностям из множества FSE и соответствующим им сущностям-образам и не может реализовать доступы к любым сущностям или информационные потоки.

Условие 3. Из потенциального доверенного субъекта из множества PS может быть создан только доверенный субъект из множества FSS. При этом множество PS не изменяется в процессе функционирования системы.

Условие 4. Для каждого доверенного субъекта из множества FSS или потенциального доверенного субъекта из множества PS множество параметрически ассоциированных с ним сущностей не пусто (для каждого $s \in FSS \cup PS$ справедливо неравенство $]s[\neq \varnothing$). Для каждого доверенного субъекта из множества FSS множество функционально ассоциированных с ним сущностей состоит только из самого субъекта (для каждого $s \in FSS$ справедливо равенство $[s] = \{s\}$), и невозможно получение к нему права доступа владения с использованием реализованного к нему информационного потока по памяти.

Условие 5. Для каждой сущности из множества FSE существует доверенный субъект из множества FSS или потенциальный доверенный субъект из множества PS, обладающий правами доступа на чтение и запись к сущности и к соответствующей ей сущности-образу (для каждой $e \in FSE$ существует субъект $s \in FSS \cup PS$, обладающий правами доступа $(s, e, read_r)$, $(s, e, write_r)$, $(s, fs(e), read_r)$, $(s, fs(e), write_r)$).

Условие 6. Каждый доверенный субъект, не входящий во множество FSS, обладает всеми правами доступа ко всем сущностям, не входящим во множества FSE, FSS и $\{e \in E$: существует $s \in S$ и $e \in]s[\}$ (множество сущностей, параметрически ассоциированных с субъектами).

В рамках предположений 1 и 2 дадим определение состояния системы.

Определение 3. Пусть определены множества S, PS, E, $R \subseteq (S \cup PS) \times E \times R_r$, $A \subseteq S \times E \times R_a$, $F \subseteq E \times E \times R_f$ и функция иерархии сущностей H. Определим $G{=}\ (S, E, R \cup A \cup F, H)$ — конечный помеченный ориентированный граф, без петель, где элементы множеств S, PS, E являются вершинами графа, элементы множества $R \cup A \cup F$ — ребрами графа. Назовем $G{=}\ (S, E, R \cup A \cup F, H)$ графом прав доступа, доступов и информационных потоков или, сокращенно, графом доступов. При этом в графе доступов будем использовать следующие обозначения:

— вершины из множества S (соответствующие субъектам) в графе доступов будут обозначаться «•»;
— вершины из множества PS (соответствующие потенциальным субъектам) в графе доступов будут обозначаться «○»;
— вершины из множества $E \setminus S$ (соответствующие сущностям, не являющимся субъектами) в графе доступов будут обозначаться «⊗»,
— каждое ребро графа доступов помечено одним из элементов множества $R_r \cup R_a \cup R_f$;
— каждое ребро из множества R будет обозначаться стрелкой вида, представленного на рис. 1,*а*;
— каждое ребро из множества A будет обозначаться стрелкой вида, представленного на рис. 1,*б*;
— каждое ребро из множества F, помеченное $write_m$, будет обозначаться стрелкой вида, представленного на рис. 1,*в*.

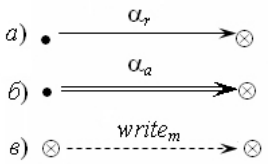


Рис. 1. Обозначения ребер графа доступов: *а* — ребро из множества R, помеченное $\alpha_r \in R_r$; *б* — ребро из множества A, помеченное $\alpha_a \in R_a$; *в* — ребро из множества F, помеченное $write_m$

Используем также обозначения:
$\Sigma(G^*, OP)$ — система, при этом:

— каждое состояние системы представляется графом доступов;
— G^* — множество всех возможных состояний;
— OP — множество правил преобразования состояний, определенных в таблице.

$G \vdash_{op} G'$ — 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.

При анализе правил преобразования состояний и траекторий функционирования системы, в результате реализации которых возникают запрещенные информационные потоки по памяти, применим подход, аналогичный использованному в рамках ФАС ДП-модели и ФПАС ДП-модели. Кроме того, в соответствии с условием 7 предположения 1 доверенные субъекты, не входящие во множество *FSS*, в процессе функционирования системы не реализуют новые доступы к сущностям. В противном случае, любой недоверенный субъект с помощью доверенных субъектов мог бы реализовать к себе информационный поток по памяти от любой сущности, защищенной ФС, для которой в системе существует соответствующий ей доверенный субъект из множества *FSS*. Таким образом, будем считать, что в дальнейшем выполняется следующее предположение.

Предположение 3. В процессе функционирования системы доверенные субъекты:

- не дают недоверенным субъектам права доступа к сущностям;
- не берут у недоверенных субъектов права доступа к сущностям;
- не получают прав доступа владения к другим субъектам;
- не создают доверенных субъектов из потенциальных доверенных субъектов.

Кроме того, доверенные субъекты, не входящие во множество *FSS*, не реализуют новые доступы к сущностям.

На основе предположения 3 дадим определение.

Определение 4. Назовем траекторию функционирования системы $\Sigma(G^*, OP)$ траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных потоков по памяти, если при ее реализации используются монотонные правила преобразования состояний, и:

- доверенные субъекты не инициируют выполнения следующих правил преобразования состояний: $take\_right(\alpha_r, u, x, e)$, $grant\_right(\alpha_r, u, x, e)$, $create\_entity(x, y, z)$, $create\_subject(x, y, z)$, $potential\_subject(u, ps, y)$, $control(u, y, e)$, $know(u, y)$, $access\_read(u, e)$, $access\_write(u, e)$;
- доверенные субъекты могут инициировать выполнение правил преобразования состояний: $own\_take(\alpha_r, u, e)$, $create\_entity(x, y, z)$, $create\_subject(x, y, z)$, $find(u, e, e')$, $post(u, e, e')$, $pass(u, e, e')$,

где $u, y \in L_S$ — доверенные субъекты, $x \in N_S$ — недоверенный субъект, $ps \in PS$ — потенциальный доверенный субъект, e, e' — сущности, $\alpha_r \in R_r$ — право доступа.

Таким образом, в рамках предположений 1 и 2 в дальнейшем будем рассматривать траектории без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных потоков по памяти. При этом по сравнению с ФАС ДП-моделью и ФПАС ДП-моделью в ФС ДП-модели (таблица):

- заданы без использования информационных потоков по времени, права доступа $append_r$ и доступа $append_a$ условия и результаты применения монотонных правил преобразования состояний: $take\_right(\alpha_r, x, y, z)$, $grant\_right(\alpha_r, x, y, z)$, $own\_take(\alpha_r, x, y)$, $create\_entity(x, y, z)$, $create\_subject(x, y, z)$, $control(x, y, z)$, $know(x, y)$, $access\_read(x, y)$, $access\_write(x, y)$, $find(x, y, z)$, $post(x, y, z)$, $pass(x, y, z)$;

— для обеспечения возможности создания доверенными субъектами информационных потоков по памяти при наличии у них доступов к сущностям изменены условия применения правил *find*(x, y, z), *post*(x, y, z), *pass*(x, y, z);
— не рассматриваются правила *flow*(x, y, y', z) и *rename_entity*(x, y, z), так как в результате их применения только информационные потоки по времени;
— не рассматривается правило *access_append*(x, y), так как оно используется для получения к сущности доступа $append_a$ с применением права доступа $append_r$;
— добавлено новое правило *potential_subject*(x, y, z), позволяющее недоверенному субъекту создать доверенного субъекта из потенциального доверенного субъекта.

Правила преобразования состояний ФС ДП-модели

Правило	Исходное состояние $G = (S, E, R \cup A \cup F, H)$	Результирующее состояние $G' = (S', E', R' \cup A' \cup F', H')$
1	2	3
take_right(α_r, x, y, z)	$x \in N_S \cap S$, $y \in S$, $z \in E \setminus FSE$, $x \neq z$, $\alpha_r \in R_r$, $(x, y, own_r) \in R$, $(y, z, \alpha_r) \in R$	$S' = S$, $E' = E$, $A' = A$, $H' = H$, $F' = F$, $R' = R \cup \{(x, z, \alpha_r)\}$
grant_right(α_r, x, y, z)	$x \in N_S \cap S$, $y \in S$, $z \in E \setminus FSE$, $y \neq z$, $\alpha_r \in R_r$, $(x, y, own_r) \in R$, $(x, z, \alpha_r) \in R$	$S' = S$, $E' = E$, $A' = A$, $H' = H$, $F' = F$, $R' = R \cup \{(y, z, \alpha_r)\}$
own_take(α_r, x, y)	$x \in S$, $y \in E$, $\alpha_r \in R_r$, $(x, y, own_r) \in R$	$S' = S$, $E' = E$, $A' = A$, $H' = H$, $F' = F$, $R' = R \cup \{(x, y, \alpha_r)\}$
create_entity(x, y, z)	$x \in S$, $y \notin E$, $z \in E \setminus S$, $(x, z, write_r) \in R$	$S' = S$, $E' = E \cup \{y\}$, $A' = A$, $F' = F$, $H'(z) = H(z) \cup \{y\}$, $H'(y) = \varnothing$, для $e \in E \setminus \{z\}$ выполняется равенство $H'(e) = H(e)$, $R' = R \cup \{(x, y, own_r)\}$
create_subject(x, y, z)	$x \in S$, $y \in E$, $z \notin E$, $(x, y, execute_r) \in R$	$S' = S \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $F' = F$, $H'(x) = H(x) \cup \{z\}$, $H'(z) = \varnothing$, для $e \in E \setminus \{x\}$ выполняется равенство $H'(e) = H(e)$, $R' = R \cup \{(x, z, own_r)\}$
potential_subject(x, y, z)	$x \in N_S \cap S$, $y \in PS$, $z \notin E$, и для каждой $e \in E$ такой, что $e \in]y[$, существует $(e, x, write_m) \in F$	$S' = S \cup \{z\}$, $FSS' = FSS \cup \{z\}$, $E' = E \cup \{z\}$, $A' = A$, $F' = F$, $H'(x) = H(x) \cup \{z\}$, $H'(z) = \varnothing$, для $e \in E \setminus \{x\}$ выполняется равенство $H'(e) = H(e)$, $R' = R \cup \{(x, z, own_r)\} \cup \{(z, e, \alpha_r)$: $(y, e, \alpha_r) \in R\}$

Продолжение таблицы

1	2	3
$know(x, y)$	$x \in N_S \cap S$, $y \in S$, $x \neq y$, и для каждой $e \in E$ такой, что $e \in]y[$, существует $(e, x, write_m) \in F$	$S' = S$, $E' = E$, $A' = A$, $H' = H$, $F' = F$, $R' = R \cup \{(x, y, own_r)\}$
$control(x, y, z)$	$x \in N_S \cap S$, $y \in S$, $x \neq y$, $z \in E$, $z \in [y]$ и или $x = z$, или $(x, z, write_m) \in F$	$S' = S$, $E' = E$, $A' = A$, $H' = H$, $F' = F$, $R' = R \cup \{(x, y, own_r)\}$
$access\_write(x, y)$	$x \in FSS \cup (N_S \cap S)$, $y \in E$, $(x, y, write_r) \in R$	$S' = S$, $E' = E$, $R' = R$, $H' = H$, $A' = A \cup \{(x, y, write_a)\}$, $F' = F \cup \{(x, y, write_m)\}$
$access\_read(x, y)$	$x \in FSS \cup (N_S \cap S)$, $y \in E$, $(x, y, read_r) \in R$	$S' = S$, $E' = E$, $R' = R$, $H' = H$, $A' = A \cup \{(x, y, read_a)\}$, $F' = F \cup \{(y, x, write_m)\}$
$control(x, y, z)$	$x \in N_S \cap S$, $y \in S$, $x \neq y$, $z \in E$, $z \in [y]$ и или $x = z$, или $(x, z, write_m) \in F$	$S' = S$, $E' = E$, $A' = A$, $H' = H$, $F' = F$, $R' = R \cup \{(x, y, own_r)\}$
$find(x, y, z)$	$x, y \in S$, $z \in E$, $x \neq z$, и либо $x = y$, $x \in L_S \cap S$ и $(x, z, write_a) \in A$, либо $x \neq y$ и $\{(x, y, \alpha), (y, z, \beta)\} \subset R \cup A \cup F$, где если $x \in L_S \cap S$, то $\alpha \in \{write_a, write_m\}$, если $x \in N_S \cap S$, то $\alpha \in \{write_r, write_m\}$, если $y \in L_S \cap S$, то $\beta = \{write_a, write_m\}$, если $y \in N_S \cap S$, то $\beta \in \{write_r, write_m\}$	$S' = S$, $E' = E$, $R' = R$, $A' = A$, $H' = H$, $F' = F \cup \{(x, z, write_m)\}$
$post(x, y, z)$	$x, z \in S$, $y \in E$, $x \neq z$, $\{(x, y, \alpha), (z, y, \beta)\} \subset R \cup A \cup F$, где если $x \in L_S \cap S$, то $\alpha \in \{write_a, write_m\}$, если $x \in N_S \cap S$, то $\alpha \in \{write_r, write_m\}$, если $z \in L_S \cap S$, то $\beta = read_a$, если $z \in N_S \cap S$, то $\beta = read_r$	$S' = S$, $E' = E$, $R' = R$, $A' = A$, $H' = H$, $F' = F \cup \{(x, z, write_m)\}$

Окончание таблицы

1	2	3
$pass(x, y, z)$	$y \in S$, $x, z \in E$, $x \neq z$, и либо $y = z$, $y \in L_S \cap S$ и $(y, x, read_a) \in A$, либо $y \neq z$ и $\{(y, x, \beta), (y, z, \alpha)\} \subset R \cup A \cup F$, где если $y \in L_S \cap S$, то $\alpha \in \{write_a, write_m\}$, $\beta = read_a$, если $y \in N_S \cap S$, то $\alpha \in \{write_r, write_m\}$, $\beta = read_r$	$S' = S$, $E' = E$, $R' = R$, $A' = A$, $H' = H$, $F' = F \cup \{(x, z, write_m)\}$

Правило преобразования состояний $potential\_subject(x, y, z)$ позволяет недоверенному субъекту x, реализовавшему к себе информационные потоки по памяти от всех сущностей, параметрически ассоциированных с потенциальным доверенным субъектом y, создать соответствующего y доверенного субъекта z (рис. 2). При этом субъект z получает все права доступа субъекта y, субъект x получает доступ владения own_r к субъекту z.

При анализе условий реализации запрещенных информационных потоков по памяти будем использовать следующие определения.

Определение 5. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — начальное состояние системы $\Sigma(G^*, OP, G_0)$. Запрещенным информационным потоком по памяти является информационный поток от сущности $e \in FSE$, защищенной ФС, к недоверенному субъекту $x \in N_S \cap S_0$ в случае, когда в начальном состоянии G_0 субъект x не имеет прав доступа $read_r$ или own_r к сущности-образу $fs(e) \in E$, соответствующей сущности e.

Определение 6. В рамках ФС ДП-модели будем говорить, что система $\Sigma(G^*, OP, G_0)$ является безопасной в случае, когда невозможен переход системы в состояние, в котором реализуется запрещенный информационный поток по памяти, удовлетворяющий условиям определения 5.

Определение 7. Нарушитель в рамках ФС ДП-модели — любой недоверенный субъект системы.

Примеры разрешенных и запрещенных информационных потоков по памяти приведены на рис. 3.

Анализ траекторий системы без получения недоверенными субъектами прав доступа владения к доверенным субъектам.

Рассмотрим частный случай, когда при реализации запрещенных информационных потоков по памяти недоверенные субъекты не применяют правила вида $control(x, y, z)$, $know(x, y)$ или $potential\_subject(x, y, z)$ для получения прав доступа владения к доверенным субъектам системы. Дадим определение.

Определение 8. Назовем траекторию функционирования системы $\Sigma(G^*, OP)$ траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам, если она является траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных

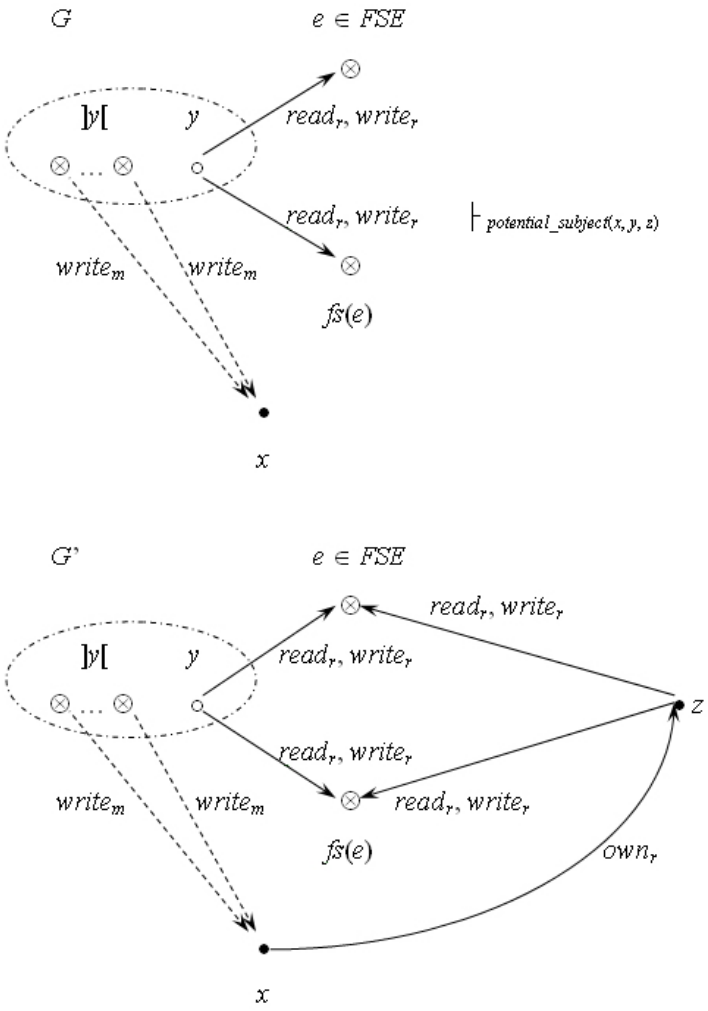


Рис. 2. Пример применения правила *potential_ subject*(x, y, z)

потоков по памяти, и при ее реализации недоверенные субъекты не инициируют выполнение правил вида *control*(x, y, z), *know*(x, y) и *potential_ subject*(x, y, z).

Определение 9. Назовем состояние G системы $\Sigma(G^*, OP)$ безопасным относительно прав доступа, когда в нем недоверенные субъекты не обладают правами доступа к доверенным субъектам.

Определение 10. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$ и сущности $x, y \in E_0$, где $x \neq y$. Определим предикат *simple_ can_ write_ memory*(x, y, G_0), который будет истинным тогда и только тогда, когда 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 $op_1, \ldots, op_N$, где $N \geqslant 0$, такие, что траектория $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(x, y, write_m) \in F_N$.

В рамках ФС ДП-модели с учетом предположений 1–3 воспользуемся определенными и обоснованными в БК ДП-модели и классической модели *Take-Grant* необходимыми и достаточными условиями истинности предиката *can_ share*(α, x, y, G_0). При этом доверенные субъекты с учетом предположения 3 могут рассматриваться как объекты модели *Take-Grant*.

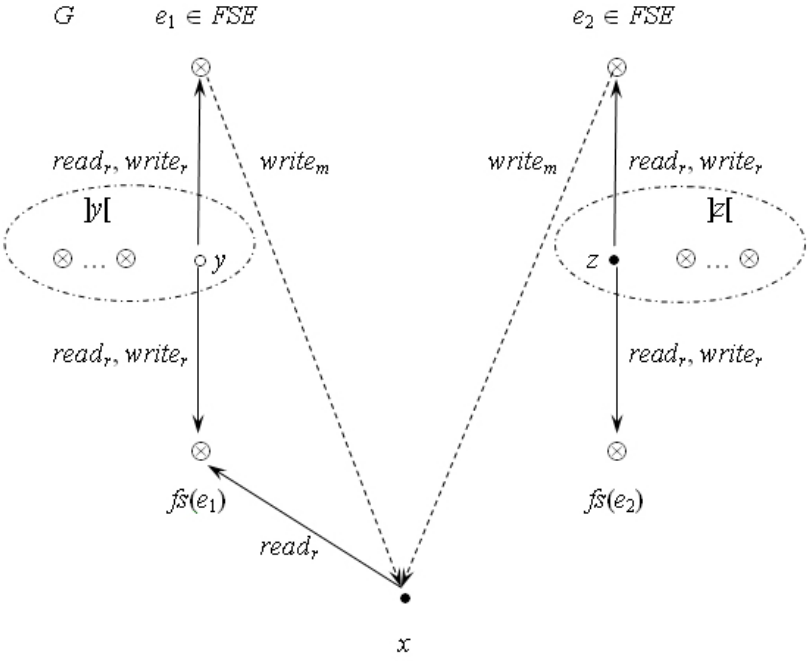


Рис. 3. Примеры информационных потоков по памяти: $(e_1,\ x,\ write_m)$ — разрешенный информационный поток; $(e_2,\ x,\ write_m)$ — запрещенный информационный поток

Определение 11. Пусть $G_0 = (S_0,\ E_0,\ R_0 \cup A_0 \cup F_0,\ H_0)$ — состояние системы $\Sigma(G^*,\ OP)$, в котором существуют субъект $x \in S_0$ и сущность $y \in E_0$, где $x \neq y$, и пусть право доступа $\alpha \in R_r$. Определим предикат $simple\_can\_share(\alpha,\ x,\ y,\ G_0)$, который будет истинным тогда и только тогда, когда 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 $op_1,\ \ldots,\ op_N$, где $N \geqslant 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам, и $(x,\ y,\ \alpha) \in R_N$.

Определение 12. Мостом в состоянии G между двумя недоверенными субъектами называется путь в графе-состоянии, удовлетворяющий одному из условий:

— субъекты соединены ребром, без учета направления помеченным правом доступа own_r;

— субъекты соединены путем, проходящим через доверенных субъектов, словарная запись которого имеет вид: $\overrightarrow{own_r}^*\ \overleftarrow{own_r}^*$, где символ «*» означает многократное, в том числе нулевое, повторение.

Определение 13. Пролетом моста в состоянии G называется путь с началом в недоверенном субъекте и концом в доверенном субъекте, проходящий через доверенных субъектов, словарная запись которого имеет вид: $\overrightarrow{own_r}^*$, где символ «*» означает многократное повторение.

Определение 14. Два недоверенных субъекта x и y в состоянии G являются own-связанными, когда существует последовательность недоверенных субъектов $s_1,\ \ldots,\ s_n$, где $n \geqslant 2$, таких, что $s_1 = x$, $s_n = y$, и каждая пара $s_i,\ s_{i+1}$ соединена мостом, где $1 \leqslant i < n$.

Утверждение 8. Пусть $G_0 = (S_0,\ E_0,\ R_0 \cup A_0 \cup F_0,\ H_0)$ — состояние системы $\Sigma(G^*,\ OP)$, в котором существуют субъект $x \in S_0$, сущность $y \in E_0$, где $x \neq y$, и пусть право доступа $\alpha \in R_r$. Предикат $simple\_can\_share(\alpha,\ x,\ y,\ G_0)$ является истинным тогда и только тогда, когда выполняются условия.

Условие 1. Существует субъект $s \in S_0$ такой, что или $(s,\ y,\ \alpha) \in R_0$, или $(s,\ y,\ own_r) \in R_0$.

Условие 2. Существуют недоверенные субъекты $x',\ s' \in N_S \cap S_0$ такие, что выполняются условия:

— или $x = x'$, или x' соединен с x пролетом моста;
— или $s = s'$, или s' соединен с s пролетом моста;
— x' и s' являются *own*-связанными.

Доказательство. Справедливость данного утверждения обосновывается в рамках классической модели *Take-Grant*. ■

Определим и обоснуем алгоритмически проверяемые необходимые и достаточные условия истинности предиката $simple\_can\_write\_memory(x,\ y,\ G_0)$.

Теорема 1. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G*, OP)$ и сущности $x,\ y \in E_0$, где $x \neq y$. Предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ является истинным тогда и только тогда, когда существует последовательность сущностей $e_1, \ldots, e_m \in E_0$, где $e_1 = x$, $e_m = y$ и $m \geqslant 2$, таких, что для каждого i =1, ..., $m-1$ выполняется одно из условий.

Условие 1. $e_i \in L_S \cap S_0$ и или $(e_i,\ e_{i+1},\ write_m) \in F_0$, или $(e_i,\ e_{i+1},\ write_a) \in A_0$.

Условие 2. $e_i \in FSS_0 \cup (N_S \cap S_0)$ и или $(e_i,\ e_{i+1},\ write_m) \in F_0$, или истинен предикат $simple\_can\_share(write_r,\ e_i,\ e_{i+1},\ G_0)$.

Условие 3. $e_{i+1} \in L_S \cap S_0$ и $(e_{i+1},\ e_i,\ read_a) \in A_0$.

Условие 4. $e_{i+1} \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(read_r,\ e_{i+1},\ e_i,\ G_0)$.

Условие 5. $e_i \in N_S \cap S_0$, $e_{i+1} \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(own_r,\ e_i,\ e_{i+1},\ G_0)$.

Условие 6. $e_{i+1} \in N_S \cap S_0$, $e_i \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(own_r,\ e_{i+1},\ e_i,\ G_0)$.

Доказательство. Докажем достаточность выполнения условий теоремы для истинности предиката $simple\_can\_write\_memory(x,\ y,\ G_0)$.

Пусть существует последовательность сущностей $e_1, \ldots, e_m \in E_0$, где $e_1 = x$, $e_m = y$ и $m \geqslant 2$, таких, что выполняются условия теоремы. Выполним доказательство индукцией по длине m последовательности сущностей.

Пусть m= 2. Возможны шесть случаев.

Первый случай: $x \in L_S \cap S_0$ и или $(x,\ y,\ write_m) \in F_0$, или $(x,\ y,\ write_a) \in A_0$. Если $(x,\ y,\ write_m) \in F_0$, то предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен. Если $(x, y, write_a) \in A_0$, то положим $op_1 = find(x,\ x,\ y)$, $G_0 \vdash_{op_1} G_1$. Тогда $(x,\ y,\ write_m) \in F_1$ и предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен.

Второй случай: $x \in FSS_0 \cup (N_S \cap S_0)$ и или $(x,\ y,\ write_m) \in F_0$, или истинен предикат $simple\_can\_share(write_r,\ x,\ y,\ G_0)$. Если $(x,\ y,\ write_m) \in F_0$, то предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен. Если истинен предикат $simple\_can\_share(write_r,\ x,\ y,\ G_0)$, то по определению 11 существуют состояния G_1, ..., G_N и правила преобразования состояний op_1, ..., op_N, где $N \geqslant 0$, такие, что

$G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(x,\ y,\ write_r) \in R_N$. Пусть $op_{N+1} = access\_write(x,\ y)$ и $G_N \vdash_{op_{N+1}} G_{N+1}$. Тогда $(x,\ y,\ write_m) \in F_{N+1}$ и предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен.

Третий случай: $y \in L_S \cap S_0$ и $(y,\ x,\ read_a) \in A_0$. Положим $op_1 = pass(x,\ y,\ y)$, $G_0 \vdash_{op_1} G_1$. Тогда $(x,\ y,\ write_m) \in F_1$ и предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен.

Четвертый случай: $y \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(read_r, y, x, G_0)$. Тогда по определению 11 существуют состояния $G_1,\ \ldots,\ G_N$ и правила преобразования состояний $op_1, \ldots, op_N$, где $N \geqslant 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(y, x, read_r) \in R_N$. Пусть $op_{N+1} = access\_read(y, x)$ и $G_N \vdash_{op_{N+1}} G_{N+1}$. Тогда $(x, y, write_m) \in F_{N+1}$ и предикат $simple\_can\_write\_memory(x, y, G_0)$ истинен.

Пятый случай: $x \in N_S \cap S_0$, $y \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(own_r, x, y, G_0)$. По определению 11 существуют состояния $G_1, \ldots, G_N$ и правила преобразования состояний $op_1, \ldots, op_N$, где $N \geqslant 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(x, y, own_r) \in R_N$. Воспользуемся предположением базовой ДП-модели, согласно которому для любого субъекта в любом состоянии системы существует сущность-контейнер, в составе которой он может создать новую сущность. Следовательно, существует сущность-контейнер $e \in E_N$, в составе которой субъект x может создать новую сущность. Пусть $op_{N+1} = create\_entity(x, z, e)$, $op_{N+2} = own\_take(write_r, x, z)$, $op_{N+3} = own\_take(read_r, x, z)$, $op_{N+4} = grant\_right(read_r, x, y, z)$, $op_{N+5} = access\_read(y, z)$, $op_{N+6} = post(x, z, y)$, и $G_N \vdash_{op_{N+1}} G_{N+1} \vdash_{op_{N+2}} \ldots \vdash_{op_{N+6}} G_{N+6}$. Тогда $(x, y, write_m) \in F_{N+6}$ и предикат $simple\_can\_write\_memory(x, y, G_0)$ истинен. На рис. 4 приведена последовательность преобразований состояний, при этом показаны только ребра графов-состояний, которые необходимы для применения правил.

Шестой случай: $y \in N_S \cap S_0$, $x \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(own_r,\ y,\ x,\ G_0)$. Доказательство для шестого случая осуществляется аналогично доказательству для пятого случая.

Докажем индуктивный шаг. Пусть $m > 2$ и утверждение теоремы верно для всех последовательностей сущностей длины $k < m$. Докажем, что утверждение теоремы верно для всех последовательностей сущностей длины m.

Пусть $e_1,\ \ldots,\ e_m \in E_0$ — последовательность сущностей, где $e_1 = x$, $e_m = y$. Возможны два случая: $x \in S_0$ или $x \in E_0 \setminus S_0$.

Первый случай: $x \in S_0$. Положим $e_{m-1} = z$. Тогда по предположению индукции существуют состояния $G_1,\ \ldots,\ G_N$ и правила преобразования состояний $op_1,\ \ldots,\ op_N$, где $N \geqslant 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(x,\ z,\ write_m) \in F_N$.

Если $z \in S_0$, то по предположению индукции существуют состояния $G_{N+1}, \ldots, G_{N+K}$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, где $K \geqslant 0$, такие, что $G_N \vdash_{op_{N+1}} G_{N+1} \vdash_{op_{N+2}} \ldots \vdash_{op_{N+K}} G_{N+K}$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(z,\ y,\ write_m) \in F_{N+K}$. Пусть $op_{N+K+1} = find(x,\ z,\ y)$, тогда $G_{N+K} \vdash_{op_{N+K+1}} G_{N+K+1}$ и $(x, y, write_m) \in F_{N+K+1}$, следовательно, предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен.

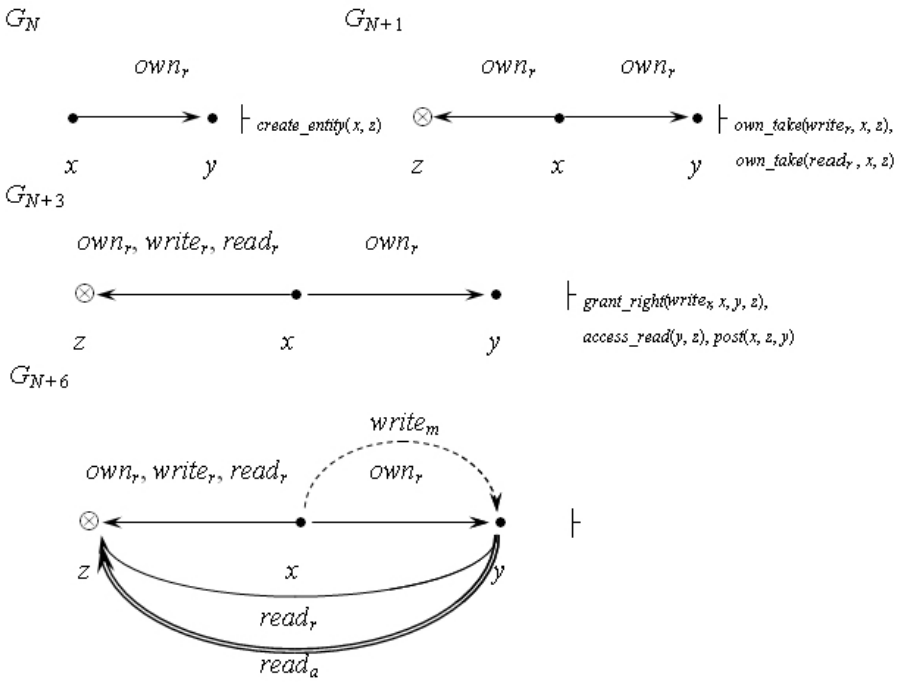


Рис. 4. Случай $x \in N_S \cap S_0$, $y \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $can\_share(own_r, x, y, G_0)$

Если $z \in E_0 \backslash S_0$, то выполняется одно из следующих условий:

— $y \in L_S \cap S_0$ и $(y, z, read_a) \in A_0$;
— $y \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(read_r, y, z, G_0)$.

Если $y \in L_S \cap S_0$ и $(y, z, read_a) \in A_0$, то пусть $M = N$.

Если $y \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(read_r, y, z, G_0)$, то по определению 11 существуют состояния $G_{N+1}, \ldots, G_{N+K}$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, где $K \geqslant 0$, такие, что $G_N \vdash_{op_{N+1}} G_{N+1} \vdash_{op_{N+2}} \ldots \vdash_{op_{N+K}} G_{N+K}$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(y, z, read_r) \in R_{N+K}$. Пусть $op_{N+K+1} = access\_read(y, z)$ и $M = N + K + 1$.

Положим $op_{M+1} = post(x, z, y)$, $G_M \vdash_{op_{M+1}} G_{M+1}$. Тогда $(x, y, write_m) \in F_{M+1}$ и предикат $simple\_can\_write\_memory(x, y, G_0)$ истинен.

Второй случай: $x \in E_0 \backslash S_0$. Положим $e_2 = z$. Тогда по условию теоремы $z \in S_0$ и по предположению индукции существуют состояния $G_1, \ldots, G_N$ и правила преобразования состояний $op_1, \ldots, op_N$, где $N \geqslant 0$, такие, что $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(z, y, write_m) \in F_N$. При этом выполняется одно из следующих условий:

— $z \in L_S \cap S_0$ и $(z, x, read_a) \in A_0$;
— $z \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(read_r, z, x, G_0)$.

Если $z \in L_S \cap S_0$ и $(z, x, read_a) \in A_0$, то пусть $M = N$.

Если $z \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $simple\_can\_share(read_r, z, x, G_0)$, то по определению 11 существуют состояния $G_{N+1}, \ldots, G_{N+K}$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, где $K \geqslant 0$, такие, что $G_N \vdash_{op_{N+1}} G_{N+1} \vdash_{op_{N+2}} \ldots \vdash_{op_{N+K}} G_{N+K}$ является траекторией без получения недоверенными субъектами прав

доступа владения к доверенным субъектам и $(z,\ x,\ read_r) \in R_{N+K}$. Пусть $op_{N+K+1} = access\_read(z,\ x)$ и $M = N + K + 1$.

Положим $op_{M+1} = pass(x,\ z,\ y)$, $G_M \vdash_{op_{M+1}} G_{M+1}$. Тогда $(x,\ y,\ write_m) \in F_{M+1}$ и предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$ истинен.

Индуктивный шаг доказан. Доказательство достаточности выполнения условий теоремы для истинности предиката $simple\_can\_write\_memory(x,\ y,\ G_0)$ выполнено.

Докажем необходимость выполнения условия теоремы для истинности предиката $simple\_can\_write\_memory(x,\ y,\ G_0)$.

Пусть истинен предикат $simple\_can\_write\_memory(x,\ y,\ G_0)$, при этом по определению 10 существуют состояния $G_1,\ \ldots,\ G_N$ и правила преобразования состояний $op_1,\ \ldots,\ op_N$, где $N \geqslant 0$, такие, что траектория $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без получения недоверенными субъектами прав доступа владения к доверенным субъектам и $(x,\ y,\ write_m) \in F_N$. Среди всех этих последовательностей выберем ту, у которой длина N является минимальной. Проведем доказательство индукцией по длине N последовательности преобразований состояний.

Пусть $N = 0$, тогда $(x,\ y,\ write_m) \in F_0$, $m{=}\ 2$ и условие 1 или 2 теоремы выполнено.

Пусть $N > 0$ и утверждение теоремы верно для всех последовательностей преобразований состояний длины $l < N$. Тогда $(x,\ y,\ write_m) \notin F_{N-1}$ и существует правило преобразования состояний op_N такое, что $G_{N-1} \vdash_{op_N} G_N$ и $(x,\ y,\ write_m) \in F_N$.

Из определения правил преобразования состояний следует, что возможны семь случаев:

— $x \in FSS_0 \cup (N_S \cap S_0)$, $(x, y, write_r) \in R_{N-1}$, $op_N = access\_write(x, y)$;
— $y \in FSS_0 \cup (N_S \cap S_0)$, $(y, x, read_r) \in R_{N-1}$ и $op_N = access\_read(y, x)$;
— $x \in L_S \cap S_0$, $(x, y, write_a) \in A_{N-1}$, $op_N = find(x, x, y)$;
— $x \in S_0$ и существует субъект $z \in S_{N-1}$ такой, что $op_N = find(x, z, y)$, $\{(x, z, \alpha), (z, y, \beta)\} \subset R_{N-1} \cup A_{N-1} \cup F_{N-1}$, где если $x \in L_S \cap S_0$, то $\alpha \in \{write_a, write_m\}$, если $x \in N_S \cap S_0$, то $\alpha \in \{write_r, write_m\}$, если $z \in L_S \cap S_{N-1}$, то $\beta = \{write_a, write_m\}$, если $z \in N_S \cap S_{N-1}$, то $\beta \in \{write_r, write_m\}$;
— $x, y \in S_0$ и существует сущность $z \in E_{N-1}$ такая, что $op_N = post(x, z, y)$, $\{(x, z, \alpha), (y, z, \beta)\} \subset R_{N-1} \cup A_{N-1} \cup F_{N-1}$, где если $x \in L_S \cap S_0$, то $\alpha \in \{write_a, write_m\}$, если $x \in N_S \cap S_0$, то $\alpha \in \{write_r, write_m\}$, если $y \in L_S \cap S_0$, то $\beta = read_a$, если $y \in N_S \cap S_0$, то $\beta = read_r$;
— $y \in L_S \cap S_0$, $(y, x, read_a) \in A_{N-1}$, $op_N = pass(x, y, y)$;
— $x, y \in E_0$ и существует субъект $z \in S_{N-1}$, $op_N = pass(x, z, y)$, $\{(z, x, \beta), (z, y, \alpha)\} \subset R_{N-1} \cup A_{N-1} \cup F_{N-1}$, где если $z \in L_S \cap S_{N-1}$, то $\alpha \in \{write_a, write_m\}$, $\beta = read_a$, если $z \in N_S \cap S_{N-1}$, то $\alpha \in \{write_r, write_m\}$, $\beta = read_r$.

В первом случае истинен предикат $simple\_can\_share(write_r,\ x,\ y,\ G_0)$, и условие 2 теоремы выполнено.

Во втором первом случае истинен предикат $simple\_can\_share(read_r,\ y,\ x,\ G_0)$, и условие 4 теоремы выполнено.

Третий случай: $x \in L_S \cap S_0$, $(x,\ y,\ write_a) \in A_{N-1}$, $op_N = find(x,\ x,\ y)$. Предположим $(x,\ y,\ write_a) \notin A_0$, тогда по предположению 3 доверенный субъект $x \in FSS_0$ и существует $0 \leqslant M < N$ такое, что $op_M = access\_write(x,\ y)$. Следовательно, выполняется условие $(x,\ y,\ write_m) \in F_M$, противоречие с минимальностью N. Значит, $(x,\ y,\ write_a) \in A_0$, и условие 1 теоремы выполнено.

В четвертом случае $op_N = find(x,\ z,\ y)$, $\{(x,\ z,\ \alpha),\ (z,\ y,\ \beta)\} \subset R_{N-1} \cup A_{N-1} \cup F_{N-1}$, из минимальности N следует, что $z \in S_0$, и выполняются условия:

— если $x \in L_S \cap S_0$, то $\alpha \in \{write_a, write_m\}$;
— если $x \in N_S \cap S_0$, то $\alpha \in \{write_r, write_m\}$;
— если $z \in L_S \cap S_0$, то $\beta = \{write_a, write_m\}$;
— если $z \in N_S \cap S_0$, то $\beta \in \{write_r, write_m\}$.

Пусть $x \in L_S \cap S_0$. Если $(x, z, write_a) \in A_{N-1}$, то по аналогии с третьим случаем получаем, что $(x, z, write_a) \in A_0$. Если $x \in L_S \cap S_0$ и $(x, z, write_m) \in F_{N-1}$, то истинен предикат $simple\_can\_write\_memory(x, z, G_0)$ с длиной последовательности преобразований состояний меньше N. Следовательно, по предположению индукции существует последовательность сущностей, удовлетворяющая условиям теоремы.

Пусть $x \in N_S \cap S_0$. Если $(x, z, write_r) \in R_{N-1}$, то истинен предикат $simple\_can\_share(write_r, x, z, G_0)$. Если $(x, z, write_m) \in F_{N-1}$, то истинен предикат $simple\_can\_write\_memory(x, z, G_0)$ с длиной последовательности преобразований состояний меньше N. Следовательно, по предположению индукции существует последовательность сущностей, удовлетворяющая условиям теоремы.

Аналогично рассматриваются условия $z \in L_S \cap S_0$ и $z \in N_S \cap S_0$. Таким образом, объединяя последовательности сущностей для каждого из возможных сочетаний пар условий, получаем, что в четвертом случае выполняются условия теоремы.

В пятом случае существует сущность $z \in E_{N-1}$ такая, что $op_N = post(x, z, y)$, $\{(x, z, \alpha), (y, z, \beta)\} \subset R_{N-1} \cup A_{N-1} \cup F_{N-1}$, и выполняются условия:

— если $x \in L_S \cap S_0$, то $\alpha \in \{write_a, write_m\}$;
— если $x \in N_S \cap S_0$, то $\alpha \in \{write_r, write_m\}$;
— если $y \in L_S \cap S_0$, то $\beta = read_a$;
— если $y \in N_S \cap S_0$, то $\beta = read_r$.

Если $z \in E_0$, то шаг индукции обосновывается аналогично четвертому случаю. Если $z \notin E_0$, то существуют $1 \leqslant M < N$, субъект $s \in S_{M-1}$, сущность-контейнер $e \in E_{M-1}$ такие, что $op_M = create\_entity(s, z, e)$. Из минимальности N следует, что выполняются условия $s \in S_0$ и сущность-контейнер $e \in E_0$. Из всех последовательностей преобразований выберем ту, в которой $M = 1$. Рассмотрим состояние G_1, где $G_0 \vdash_{op_1} G_1$, $S_1 = S_0$, $E_1 = E_0 \cup \{z\}$, $R_1 = R_0 \cup \{(s, z, own_r)\}$, $A_1 = A_0$, $F_1 = F_0$. Так как $(x, z, \alpha) \in R_{N-1} \cup A_{N-1} \cup F_{N-1}$, то истинен предикат $simple\_can\_write\_memory(x, z, G_1)$ с длиной последовательности состояний меньше N. Таким образом, получаем, что существуют сущности $e_1, \ldots, e_k$, где $e_1 = x$, $e_k = s$ и $k \geqslant 2$, удовлетворяющие условиям теоремы в состоянии G_1, а следовательно, в состоянии G_0. Аналогично получаем, что существуют сущности $e_k, \ldots, e_{k+m}$, где $e_k = s$, $e_{k+m} = y$ и $m \geqslant 1$, удовлетворяющие условиям теоремы в состоянии G_0. Значит, в пятом случае условия теоремы выполнены.

Седьмой случай рассматривается аналогично третьему случаю.

Восьмой случай рассматривается аналогично четвертому и пятому случаям.

Индуктивный шаг доказан. Обоснована необходимость выполнения условий теоремы для истинности предиката $simple\_can\_write\_memory(x, y, G_0)$. ■

Рассмотрим условия истинности предиката $simple\_can\_write\_memory(e, x, G_0)$ для случая, когда сущность x является недоверенным субъектом и сущность e защищена ФС.

Следствие 1. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — начальное состояние системы $\Sigma(G*, OP, G_0)$, безопасное относительно прав доступа, в котором существуют сущность $e \in FSE_0$, недоверенный субъект $x \in N_S \cap S_0$. Предикат $simple\_can\_$

$\_write\_memory(e,\ x,\ G_0)$ (недоверенный субъект x пытается реализовать запрещенный информационный поток по памяти от сущности e) является истинным тогда и только тогда, когда существует доверенный субъект $s \in FSS_0$ такой, что он обладает правами доступа на чтение и запись к сущностям e и $fs(e)$, и существует последовательность сущностей $e_1, \ldots, e_m \in E_0$, где $e_1 = x$, $e_m = fs(e)$ и $m \geqslant 2$, таких, что для каждого $i = 1, \ldots, m-1$ выполняется одно из условий.

Условие 1. $e_i \in L_S \cap S_0$, $e_{i+1} \in E_0 \setminus N_S$ и или $(e_i,\ e_{i+1},\ write_m) \in F_0$, или $(e_i,\ e_{i+1},\ write_a) \in A_0$.

Условие 2. $e_i \in N_S \cap S_0$, $e_{i+1} \in E_0 \setminus L_S$ и истинен предикат $simple\_can\_share(write_r,\ e_i,\ e_{i+1},\ G_0)$.

Условие 3. $e_{i+1} \in L_S \cap S_0$, $e_i \in E_0 \setminus N_S$ и $(e_{i+1},\ e_i,\ read_a) \in A_0$.

Условие 4. $e_{i+1} \in N_S \cap S_0$, $e_i \in E_0 \setminus L_S$ и истинен предикат $simple\_can\_share(read_r,\ e_{i+1},\ e_i,\ G_0)$.

Условие 5. $e_i, e_{i+1} \in N_S \cap S_0$ и истинен предикат $simple\_can\_share(own_r, e_i, e_{i+1}, G_0)$.

Условие 6. $e_i,\ e_{i+1} \in N_S \cap S_0$ и истинен предикат $simple\_can\_share(own_r, e_{i+1}, e_i, G_0)$.

Доказательство. В соответствии с предположениями 1 и 2 только доверенные субъекты из множества FSS_0 могут обладать правами доступа к сущностям, защищенным ФС. Значит, субъекты, не входящие во множество FSS_0, могут обладать правами доступа, реализовывать информационные потоки или доступы только к сущностям-образам из множества $fs(FSE_0)$. При этом для каждой сущности из множества FSE_0 всегда существует доверенный субъект из множества FSS_0 или потенциальный доверенный субъект из множества PS_0, обладающий правами доступа на чтение и запись к сущности и к соответствующей ей сущности-образу (для каждой $e \in FSE_0$ существует субъект $s \in FSS_0 \cup PS_0$, обладающий правами доступа $(s, e, read_r)$, $(s, e, write_r)$, $(s, fs(e), read_r)$, $(s, fs(e), write_r)$).

В соответствии с предположением 1 в начальном состоянии отсутствуют доступы к недоверенным субъектам и информационные потоки с участием недоверенных субъектов.

Так как начальное состояние безопасно относительно прав доступа, то для любых доверенного субъекта $s_1 \in FSS_0$ и субъекта $s_2 \in FSS_0 \cup (N_S \cap S_0)$ предикаты $simple\_can\_share(write_r, s_1, s_2, G_0)$, $simple\_can\_share(write_r, s_2, s_1, G_0)$, $simple\_can\_share(read_r, s_1, s_2, G_0)$, $simple\_can\_share(read_r, s_2, s_1, G_0)$, $simple\_can\_share(own_r, s_1, s_2, G_0)$ и $simple\_can\_share(own_r, s_2, s_1, G_0)$ являются ложными.

Таким образом, утверждение следствия следует из теоремы 1. ■

В соответствия с утверждением следствия 1, если начальное состояние системы безопасное относительно прав доступа, то необходимым условием истинности предиката $simple\_can\_write\_memory(e,\ x,\ G_0)$, где сущность $e \in FSE_0$, недоверенный субъект $x \in N_S \cap S_0$, является наличие доверенного субъекта $s \in FSS_0$ такого, что он обладает правами доступа на чтение и запись к сущностям e и $fs(e)$.

Следствие 2. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — начальное состояние системы $\Sigma(G*, OP, G_0)$, безопасное относительно прав доступа, в котором отсутствуют доступы или информационные потоки к или от доверенных субъектов, не существует недоверенных субъектов $x \in N_S \cap S_0$ и сущностей $e \in FSE_0$ таких, что выполняются условия или $(x,\ fs(e),\ read_r) \in R_0$, или $(x,\ fs(e),\ own_r) \in R_0$. Пусть также в системе могут реализовываться только траектории без получения недоверенными субъектами прав доступа владения к доверенным субъектам. Тогда система является безопасной.

Доказательство. Утверждение следует из теоремы 1 и следствия 1. ■

Таким образом, в следствии 2 обоснованы достаточные условия безопасности системы для случая, когда в ней могут реализовываться только траектории без получения недоверенными субъектами прав доступа владения к доверенным субъектам.

Анализ траекторий системы с возможным получением недоверенными субъектами прав доступа владения к доверенным субъектам.

Рассмотрим общий случай, когда при реализации запрещенных информационных потоков по памяти недоверенные субъекты могут применять правила вида $control(x, y, z)$, $know(x, y)$ или $potential\_subject(x, y, z)$ для получения прав доступа владения к доверенным субъектам системы. Дадим определение.

Определение 15. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$ и сущности $x, y \in E_0$, где $x \neq y$. Определим предикат $can\_write\_memory(x, y, G_0)$, который будет истинным тогда и только тогда, когда 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 $op_1, \ldots, op_N$, где $N \geqslant 0$, такие, что траектория $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных потоков по памяти и $(x, y, write_m) \in F_N$.

Так как при использовании правил вида $control(x, y, z)$, $know(x, y)$ или $potential\_subject(x, y, z)$ права доступа используются для реализации информационных потоков по памяти, а информационные потоки по памяти могут быть использованы для получения прав доступа, то необходимые условия реализации запрещенных информационных потоков по памяти должны быть определены рекурсивно, что является труднореализуемой задачей. Таким образом, определим и обоснуем достаточные условия реализации запрещенных информационных потоков по памяти. Дадим определения.

Определение 16. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$ и недоверенный субъект $x \in N_S \cap S_0$, субъект $y \in S_0$, где $x \neq y$. Определим предикат $can\_share\_own(x, y, G_0, L_S)$, который будет истинным тогда и только тогда, когда 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 $op_1, \ldots, op_N$, где $N \geqslant 0$, такие, что траектория $G_0 \vdash_{op_1} G_1 \vdash_{op_2} \ldots \vdash_{op_N} G_N$ является траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных потоков по памяти и $(x, y, own_r) \in R_N$.

Определение 17. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G*, OP)$ и недоверенный субъект $x \in N_S \cap S_0$, субъект или потенциальный доверенный субъект $y \in S_0 \cup PS$, где $x \neq y$. Определим предикат $directly\_can\_share\_own(x, y, G_0)$, который будет истинным тогда и только тогда, когда существует последовательность субъектов $s_1, \ldots, s_m \in S_0 \cup PS$, где $s_1 = x$, $s_m = y$ и $m \geqslant 2$, таких, что для каждого $i = 1, \ldots, m-1$ справедливо неравенство $s_i \neq s_{i+1}$ и выполняется одно из условий:

- $s_{i+1} \notin FSS_0$ и $s_i \in [s_{i+1}]$ (каждый доверенный субъект из множества FSS_0 по предположению 2 является функционально корректным);
- истинен предикат $simple\_can\_share(own_r, s_i, s_{i+1}, G_0)$;
- $s_{i+1} \notin FSS_0$ и существует сущность $e \in [s_{i+1}]$ такая, что истинен предикат $simple\_can\_write\_memory(s_i, e, G_0)$ (по предположению 2 невозможно получение

к доверенному субъекту из множества FSS_0 права доступа владения с использованием реализованного к нему информационного потока по памяти);

— для каждой сущности $e \in]s_{i+1}[$ истинен предикат $simple\_can\_write\_memory(s_i, e, G_0)$.

Определим и обоснуем достаточные условия истинности предиката $can\_share\_own(x, y, G_0)$.

Утверждение 9. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$, пусть также существуют недоверенный субъект $x \in N_S \cap S_0$, субъект или потенциальный доверенный субъект $y \in S_0 \cup PS$, где $x \neq y$. Предикат $can\_share\_own(x, y, G_0)$ является истинным в случае, когда существует последовательность субъектов $s_1, \ldots, s_m \in S_0 \cup PS$, где $s_1 = x$, $s_m = y$ и $m \geqslant 2$, таких, что выполняется одно из условий.

Условие 1. $m = 2$ и истинен предикат $directly\_can\_share\_own(x, y, G_0)$.

Условие 2. $m > 2$ и для каждого i =1, ..., $m - 2$ выполняется одно из условий:

— $s_i, s_{i+1} \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(s_i, s_{i+1}, G_0)$, $directly\_can\_share\_own(s_{i+1}, s_{i+2}, G_0)$;

— $i < m - 2$,$s_i, s_{i+2} \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(s_i, s_{i+1}, G_0)$, $directly\_can\_share\_own(s_{i+2}, s_{i+1}, G_0)$;

— $i < m - 2$, $s_{i+1}, s_{i+2} \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(s_{i+1}, s_i, G_0)$, $directly\_can\_share\_own(s_{i+2}, s_{i+1}, G_0)$;

— $s_{i+1} \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(s_{i+1}, s_i, G_0, L_S)$, $directly\_can\_share\_own(s_{i+1}, s_{i+2}, G_0, L_S)$.

Доказательство. Доказательство теоремы выполняется аналогично доказательству в рамках ФПАС ДП-модели достаточности условий истинности предиката $can\_share\_own(x, y, G_0)$. ■

Определим и обоснуем достаточные условия истинности предиката $can\_write\_memory(x, y, G_0)$.

Теорема 2. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$ и сущности $x, y \in E_0$, где $x \neq y$. Предикат $can\_write\_memory(x, y, G_0)$ является истинным в случае, когда существует последовательность сущностей $e_1, \ldots, e_m \in E_0$, где $e_1 = x$, $e_m = y$ и $m \geqslant 2$, таких, что для каждого i =1, ..., $m - 1$ выполняется одно из условий.

Условие 1. $e_i \in L_S \cap S_0$ и или $(e_i, e_{i+1}, write_m) \in F_0$, или $(e_i, e_{i+1}, write_a) \in A_0$.

Условие 2. $e_i \in FSS_0 \cup (N_S \cap S_0)$ и или $(e_i, e_{i+1}, write_m) \in F_0$, или истинен предикат $can\_share(write_r, e_i, e_{i+1}, G_0)$.

Условие 3. $e_{i+1} \in L_S \cap S_0$ и $(e_{i+1}, e_i, read_a) \in A_0$.

Условие 4. $e_{i+1} \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $can\_share(read_r, e_{i+1}, e_i, G_0)$.

Условие 5. $e_i \in N_S \cap S_0$, $e_{i+1} \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $can\_share\_own(e_i, e_{i+1}, G_0)$.

Условие 6. $e_{i+1} \in N_S \cap S_0$, $e_i \in FSS_0 \cup (N_S \cap S_0)$ и истинен предикат $can\_share\_own(e_{i+1}, e_i, G_0)$.

Доказательство. Доказательство осуществляется аналогично обоснованию достаточности условий теоремы 1 для истинности предиката $simple\_can\_write\_memory(x, y, G_0)$. ■

Рассмотрим условия истинности предиката $can\_write\_memory(e, x, G_0)$ для случая, когда сущность x является недоверенным субъектом и сущность e защищена ФС.

Следствие 3. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$, в котором существуют сущность $e \in FSE_0$, недоверенный субъект $x \in N_S \cap S_0$, и выполняются условия сущность $fs(e) \notin FSE_0$, и она не является параметрически ассоциированной ни с одним субъектом. Пусть также существует доверенный субъект $z \in FSS_0$ такой, что он обладает правами доступа на чтение и запись к сущностям e и $fs(e)$, и существует доверенный субъект $z' \in (L_S \cap S_0) \setminus FSS_0$ такой, что истинен предикат $can\_share\_own(x, z', G_0)$. Тогда предикат $can\_write\_memory(e, x, G_0)$ является истинным.

Доказательство. Так как выполняется условие $fs(e) \notin FSE_0 \cup FSS_0 \cup \{e \in E_0$: существует $s \in S_0$ и $e \in]s[\}$, то по предположению 2 справедливо условие $(z', fs(e), read_r) \in R_0$. Следовательно, истинен предикат $can\_share(read_r, x, fs(e), G_0)$. Кроме того, по предположению 2 являются истинными предикаты $can\_share(read_r, z, e, G_0)$ и $can\_share(write_r, z, fs(e), G_0)$. Таким образом, выполнены условия теоремы 2, и истинен предикат $can\_write\_memory(e, x, G_0)$. Следствие доказано. ■

Пример выполнения условия следствия 3 приведен на рис. 5.

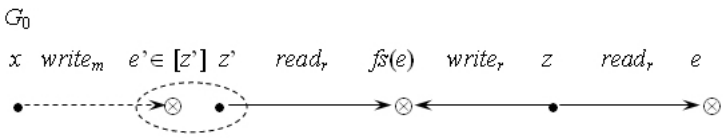


Рис. 5. Пример выполнения условия следствия 3, где $z \in FSS_0$, $e \in FSE_0$

Следствие 4. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G^*, OP)$, в котором существуют сущность $e \in FSE_0$ и недоверенный субъект $x \in N_S \cap S_0$. Пусть также существует доверенный субъект или потенциальный доверенный субъект $y \in FSS_0 \cup PS$ такой, что он обладает правами доступа на чтение и запись к сущностям e и $fs(e)$, и выполняется условие: для каждой сущности $e' \in]y[$ истинен предикат $simple\_can\_write\_memory(x, e', G_0)$. Тогда предикат $can\_write\_memory(e, x, G_0)$ является истинным.

Доказательство. По определению 17 истинен предикат $directly\_can\_share\_own(x, y, G_0)$. Следовательно, по утверждению 2 истинен предикат $can\_share\_own(x, y, G_0)$. Таким образом, выполнены условия теоремы 2 и истинен предикат $can\_write\_memory(e, x, G_0)$. Следствие доказано. ■

Рассмотрим достаточные условия, при выполнении которых является ложным предикат $can\_write\_memory(e, x, G_0)$, где сущность e защищена ФС, и субъект x является недоверенным.

Утверждение 10. Пусть $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние системы $\Sigma(G*, OP)$, в котором существуют сущность $e \in FSE_0$ и недоверенный субъект $x \in N_S \cap S_0$. Пусть также выполнены условия.

Условие 1. Не существует доверенных субъектов таких, что они обладает правами доступа на чтение и запись к сущностям e и $fs(e)$.

Условие 2. Для каждого потенциального доверенного субъекта $y \in PS$ такого, что он обладает правами доступа на чтение и запись к сущностям e и $fs(e)$, и для любого

субъекта $x' \in S_0$ существует сущность $e_y \in]y[$ такая, что выполняются условия $(e_y, x', write_m) \notin F_0$, $(x', e_y, write_a) \notin A_0$, $(x', e_y, write_r) \notin R_0$, $(x', e_y, own_r) \notin R_0$.

Тогда предикат $can\_write\_memory(e, x, G_0)$ является ложным.

Доказательство. Так как по условию 1 не существует доверенных субъектов таких, что они обладает правами доступа на чтение и запись к сущностям e и $fs(e)$, то по предположениям 1 и 2 для реализации информационного потока от сущности e необходимо создание такого доверенного субъекта $y' \in FSS$ из потенциального доверенного субъекта $y \in PS$ с использованием правила $potential\_subject(x', y, y')$, где $x' \in N_S \cap S_0$. По условию 2 не существует недоверенного субъекта $x' \in N_S \cap S_0$, удовлетворяющего условиям применения правила $potential\_subject(x', y, y')$. Следовательно, предикат $can\_write\_memory(e, x, G_0)$ является ложным. Утверждение доказано. ■

Таким образом, описаны и обоснованы достаточные условия, при выполнении которых в системе невозможна реализация запрещенного информационного потока по памяти от сущности, защищенной ФС.

ЛИТЕРАТУРА

1. *Девянин П. Н.* Анализ безопасности управления доступом и информационными потоками в компьютерных системах. М.: Радио и связь, 2006. 176 с.

2009 Математические основы компьютерной безопасности №1(3)
DOI 10.17223/20710410/3/7
УДК 681.322

ОБ ИСПОЛЬЗОВАНИИ ФОРМАЛЬНЫХ МОДЕЛЕЙ ДЛЯ АНАЛИЗА УЯЗВИМОСТЕЙ

Д. Н. Колегов

Томский государственный университет, г. Томск

E-mail: d.n.kolegov@gmail.com

В статье рассматривается возможный подход к анализу уязвимостей с использованием математических моделей безопасности компьютерных систем. В рамках ДП-моделей строится модель нарушителя, описанная в «Критериях оценки безопасности информационных технологий», и дается математическое определение стойкости к проникновению.

Ключевые слова: *анализ уязвимостей, модель нарушителя, тестирование проникновения, модели безопасности, функциональные требования, обоснование безопасности, доверие.*

Введение

Использование математических моделей является обязательным при разработке и анализе безопасности компьютерных систем (КС) с высоким уровнем доверия. Так, в соответствии с «Критериями оценки безопасности информационных технологий» [1–4] для КС с оценочным уровнем доверия (ОУД) более 5 требуется, чтобы при их разработке была использована формальная (математическая) модель политики безопасности (ADV_SPM.3) [4]. Для КС, начиная с ОУД 2, требуется обязательное проведение анализа уязвимостей и тестирования проникновения. При этом указано, что разработчик должен предоставить документацию, содержащую строгое обоснование того, что объект оценки (ОО) с идентифицированными уязвимостями является стойким к явным нападениям проникновения. Под термином «строгое обоснование» (justification) следует понимать анализ (менее строгий, чем формальный), ведущий к необходимому заключению. Начиная с ОУД 4, для КС требуется независимый анализ того, что ОО с идентифицированными уязвимостями является стойким к проникновению со стороны нарушителя, обладающего низким, умеренным или высоким потенциалом нападения (AVA_VLA). Однако требования по использованию формальных моделей для проведения анализа уязвимостей КС в [1–4] отсутствуют. В данной работе показывается возможность применения математических моделей для анализа уязвимостей КС. Для этого в рамках ДП-моделей [5, 6] описывается модель нарушителя и дается математическое определение стойкости КС к проникновению в соответствии с требованиями семейства доверия 14.4 «Анализ уязвимостей» (AVA_VLA) в [4].

1. Общие предположения и определения

Дальнейшее изложение будет вестись на основе работ [5, 6] с учетом всех определений, обозначений и теорем в них. В соответствии с [7, 8] под уязвимостью понимается некоторый недостаток КС, который может использоваться для реализации угроз. Анализ уязвимостей КС [4] включает идентификацию недостатков, внесенных на различных этапах жизненного цикла КС, и подтверждение выявленных уязвимостей посредством тестирования проникновения, позволяющим сделать заключение о возможности использования уязвимостей для нарушения безопасности. Дадим определение.

Определение 1. *Уязвимостями* будем называть сущности, функционально или параметрически ассоциированные с другими субъектами КС, если при реализации информационных потоков к ним или от них соответственно происходит нарушение безопасности. Субъекты, имеющие такие сущности, назовем *уязвимыми*.

Замечание 1. По определению 1 уязвимость является сущность-субъектом или сущность-объектом.

Определение 2. Пусть $G = (S, E, R \cup A \cup F, H)$ — состояние КС $\Sigma(G^*, OP)$. Определим элементы состояния G: E — множество сущностей; EC — множество узлов; V — множество уязвимостей; CC — множество коммуникационных каналов; IC — множество интерфейсов взаимодействия; S — множество субъектов; 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 КС иерархии подчиненности компьютеров. При этом выполняются следующие условия:

1) $V \subset E,\ CC \subset E,\ IC \subset E,\ EC \subset E$;
2) множества V, CC, EC, IC попарно не пересекаются;
3) каждая сущность $e \in E$:
   - либо является узлом КС ($e \in EC$), либо размещена на некотором единственном для каждой сущности узле (для сущности $e \in E$ существует единственный узел $c \in EC$ такой, что $e < c$),
   - либо является коммуникационным каналом ($e \in EC$),
   - либо является интерфейсом взаимодействия с КС($e \in IC$),
   - либо является уязвимостью ($e \in V$),
   - либо является пользователем или процессом пользователя ($e \in S$).

КС будем рассматривать с учетом сетевого уровня, поэтому с учетом положений БК ДП-модели [5] будем считать, что выполняется следующее предположение.

Предположение 1. Для каждого узла (компьютера) $c \in EC$ определены доверенные пользователи, обладающие правом доступа владения к каждой сущности, размещенной на данном узле, и коммуникационные каналы (сущности-объекты), через которые осуществляется передача данных между субъектами узлов КС. Если пользователь $os_c \in S$ является доверенным пользователем компьютера c или пользователь $s \in S$ обладает правом доступа владения own_r к компьютеру c в состоянии G, то он обладает правом доступа владения к каждой сущности, размещенной на данном узле. Для каждого узла определены коммуникационные каналы и интерфейсы взаимодействия (сущности-объекты), через которые осуществляется передача данных между субъектами узлов КС.

В зависимости от архитектуры системы безопасности современных КС возможно несколько способов передачи прав доступа при активизации и функционировании пользовательского процесса. Для реализаций нарушений безопасности субъектами-нарушителями используются следующие уязвимости современных КС, приводящие к получению субъектом-нарушителем прав пользователей КС:

— ошибки в ПО, реализующем прикладные и системные процессы ОС;

— ошибки в реализации, конфигурировании и использовании КС, приводящие к реализации информационных потоков по памяти и по времени между нарушителем и субъектами КС;
— возможность получения или изменения некоторых параметров КС.

Таким образом, будем использовать следующее предположение.

Предположение 2. В КС выполняются следующие условия:

— при активации субъектом-пользователем $u \in S$ некоторого процесса $p \in S$ последний наследует все права пользователя u;
— уязвимости процесса функционально-ассоциированы с субъектом-пользователем, от имени которого данный процесс запущен;
— уязвимости, связанные с раскрытием параметров функционирования КС, параметрически-ассоциированы с субъектом-пользователем, преобразования данных которого определяются этими параметрами.

2. Модель нарушителя и определение стойкости к проникновению

В семействе доверия «Анализ уязвимостей» (AVA_VLA) класса «Оценка уязвимостей» (AVA) «Критериев оценки безопасности информационных технологий» [4] определены три компоненты, включающие независимый анализ уязвимостей, проводимый оценщиком. Основная цель данного анализа — сделать заключение, что ОО является стойким к нападениям проникновения со стороны нарушителя, обладающего низким (для AVA_VLA.2), умеренным (для AVA_VLA.3) или высоким (для AVA_VLA.4) потенциалом нападения. Для достижения этой цели оценщик сначала проверяет возможности использования всех идентифицированных уязвимостей. Это осуществляется посредством тестирования проникновения. Оценщику следует принять на себя роль нарушителя с одним из указанным выше потенциалов нападения при попытке проникновения в ОО. Любое использование уязвимостей таким нарушителем оценщику следует рассматривать как «явное нападение проникновения» (в отношении элементов AVA_VLA.*.2C) в контексте компонентов AVA_VLA.2 – 4. Кроме того, при анализе уязвимостей рассматривают угрозы в предположении, что нарушитель будет в состоянии обнаружить недостатки, позволяющие получить несанкционированный доступ к ресурсам (например, данным), препятствовать выполнению функций безопасности и искажать их или же ограничивать санкционированные возможности других пользователей. Так, идентификация известных уязвимостей может быть проведена путем анализа исходного кода ПО, применения сканеров безопасности уровня узла или сети. Будем использовать следующее предположение.

Определение 3. Модель нарушителя содержит такие условия:

— существует три класса нарушителей;
— нарушитель любого класса имеет все данные о КС (например, исходные коды программного обеспечения, документация);
— нарушитель с низким потенциалом нападения инициирует выполнение всех правил преобразования в КС;
— нарушитель с умеренным потенциалом нападения может кооперироваться с некоторыми субъектами КС, кроме субъектов тестируемого узла КС;
— нарушитель с высоким потенциалом нападения может кооперироваться с любыми субъектами КС;
— нарушитель любого класса знает все уязвимости КС;

— нарушитель может осуществлять удаленный или локальный доступ к субъектам КС через коммуникационные каналы или интерфейсы взаимодействия.

С учетом «Критериев оценки безопасности информационных технологий» [4] и введенной модели нарушителя дадим следующее определение.

Определение 4. Пусть имеются $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — начальное состояние КС (G^*, OP), недоверенный субъект-нарушитель $x \in N_S \cap S_0$ и контейнер-узел $c \in EC$. Будем говорить, что узел $c \in EC$ является *стойким к проникновению нарушителя с низким потенциалом нападения* (*НПН-стойким*) тогда и только тогда, когда для любого недоверенного субъекта $y \in N_S \cap S_0$, размещенного на узле c, предикат $directly\_can\_share\_own(x, y, G_0, L_S)$ является ложным. Будем говорить, что узел $c \in EC$ является *стойким к проникновению нарушителя с умеренным потенциалом нападения* (*УПН-стойким*) тогда и только тогда, когда для любого доверенного субъекта $y \in L_S \cap S_0$, размещенного на узле c, предикат $can\_steal\_own(x, y, G_0, L_S)$ является ложным. Наконец, будем говорить, что узел $c \in EC$ является *стойким к проникновению нарушителя с высоким потенциалом нападения* (*ВПН-стойким*) тогда и только тогда, когда для любого доверенного субъекта $y \in L_S \cap S_0$, размещенного на узле c, предикат $can\_share\_own(x, y, G_0, L_S)$ является ложным.

В [5, 6] сформулированы и обоснованы условия истинности предикатов $directly\_can\_share\_own(x, y, G_0, L_S)$, $can\_share\_own(x, y, G_0, L_S)$ и $can\_steal\_own(x, y, G_0, L_S)$. Кроме того, существуют алгоритмы проверки условий истинности данных предикатов за конечное время.

Таким образом, математические модели безопасности возможно использовать не только для разработки формальных политик безопасности, но и для анализа уязвимостей КС: описания модели нарушителя, математического определения стойкости к нарушениям безопасности (проникновению), описания уязвимостей, условий их использования и передачи прав доступов и реализации информационных потоков, возникающих при этом.

ЛИТЕРАТУРА

1. Information technology — Security techniques — Evaluation criteria for IT security — Part 1: Introduction and general model. ISO/IEC 15408–1, 1999.
2. Information technology — Security techniques — Evaluation criteria for IT security — Part 2: Security functional requirements. ISO/IEC 15408–2, 1999.
3. Information technology — Security techniques — Evaluation criteria for IT security — Part 3: Security assurance requirements. ISO/IEC 15408–3, 1999.
4. Гостехкомиссия России. Руководящий документ. Безопасность информационных технологий. Критерии оценки безопасности информационных технологий. Ч. 1, 2 и 3. М., 2002.
5. *Девянин П. Н.* Анализ безопасности управления доступом и информационными потоками в компьютерных системах. М.: Радио и связь, 2006. 176 с.
6. *Колегов Д. Н.* Анализ безопасности информационных потоков по памяти в компьютерных системах с функционально и параметрически ассоциированными сущностями // Прикладная дискретная математика. 2009. № 1. С. 117–125.
7. NIST. Technical guide to information security testing and assessment. Recommendations of the National Institute of Standarts and Technology. September, 2008.
8. ФСТЭК России. Руководящий документ. Безопасность информационных технологий. Концепция оценки соответствия автоматизированных систем требованиям безопасности информации. М., 2004.

2009 Математические основы компьютерной безопасности №1(3)
DOI 10.17223/20710410/3/8
УДК 681.322

АНАЛИЗ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ПОТОКОВ ПО ПАМЯТИ В КОМПЬЮТЕРНЫХ СИСТЕМАХ С ФУНКЦИОНАЛЬНО И ПАРАМЕТРИЧЕСКИ АССОЦИИРОВАННЫМИ СУЩНОСТЯМИ

Д. Н. Колегов

Томский государственный университет, г. Томск

E-mail: d.n.kolegov@gmail.com

В статье вводится определение сущностей, параметрически-ассоциированных с субъектами компьютерных систем. Строится расширение ДП-модели, охватывающее такие сущности.

Ключевые слова: *компьютерная безопасность, математические модели безопасности, дискреционные модели, анализ безопасности, права доступа, информационные потоки.*

Введение

Одной из современных моделей анализа безопасности компьютерных систем (КС) с дискреционным управлением доступа является ДП-модель с ее расширениями [1]. Дальнейшее изложение будет вестись на основе работы [1] с учетом всех определений, обозначений и теорем в ней. Сущность называется функционально-ассоциированной с субъектом, если она определяет вид преобразования данных, выполняемого этим субъектом. В ДП-моделях с функционально-ассоциированными с субъектами сущностями (ФАС ДП-моделях) анализируется ситуация, когда реализация информационного потока по памяти к сущности, функционально-ассоциированной с субъектом, приводит к изменению вида преобразования данных, реализуемого этим субъектом.

В то же время в современных КС возможна реализация информационного потока по памяти от сущности, позволяющая получить права доступа различных субъектов, в том числе и доверенных. Такие сущности являются параметрически-ассоциироваными с субъектами КС. Например, получение субъектом-нарушителем доступа на чтение к конфигурационному файлу или реестру, в котором хранится пароль или хэш-значение пароля субъекта КС, позволяет субъекту-нарушителю получить право доступа владения к последнему субъекту. Кроме того, в настоящее время дополнительно к классическим угрозам нарушения конфиденциальности, целостности и доступности информации рассматривают угрозу раскрытия параметров КС — возможность идентификации параметров, функций безопасности и свойств КС, знание которых позволяет реализовать нарушение безопасности [2]. Например, чтение сообщения, выдаваемого субъектом-процессом при подключении к нему, позволяет нарушителю идентифицировать программное обеспечение (ПО), реализующее данный субъект-процесс КС, и получить права доступа последнего, используя известные уязвимости в ПО.

Таким образом, для обеспечения возможности анализа получения субъектом права доступа владения к другому субъекту с использованием информационного потока от сущности, параметрически-ассоциированной с последним, следует построить расширение ФАС ДП-модели, охватывающее информационные потоки указанного вида и отражающее возможность получения субъектом права доступа владения к другому субъекту в современных КС. Решение этой задачи и является целью данной работы.

Для этого вводится определение параметрически-ассоциированных сущностей с субъектами в КС, на их основе строится ДП-модель с функционально- и параметрически-ассоциированными с субъектами сущностями, которая является требуемым расширением ФАС ДП-модели, и в рамках этой модели формулируются и обосновываются необходимые и достаточные условия получения недоверенным субъектом права доступа владения к другому субъекту без кооперации с ним.

1. ДП-модель с функционально- и параметрически-ассоциированными с субъектами сущностями

Определение 1. Пусть $G = (S, E, R \cup A \cup F, H)$ — состояние КС $\Sigma(G^*, OP)$. Сущность $e \in E$ будем называть *параметрически-ассоциированной* с субъектом $s \in S$ в состоянии G, если чтение данных в сущности e субъектом $z \in S$ позволяет ему получить право владения к субъекту s в этом или последующих состояниях КС.

Замечание 1. Сущность $e \in E$, параметрически-ассоциированная с субъектом $s \in S$, содержит аргументы операций преобразования данных, выполняемого субъектом s в этом или последующих состояниях КС.

Замечание 2. В множество сущностей, параметрически-ассоциированных с субъектом $s \in S$, могут входить сущности, на которые субъект s не имеет прав доступа. Например, сущность-пароль, параметрически-ассоциированная с недоверенным субъектом-пользователем в ОС семейства UNIX, хранится в файле /etc/shadow, правами доступа к которому могут обладать только доверенные субъекты-процессы данной ОС [3]. Кроме того, возможно создание субъектов ОС, которые препятствовали бы доступу субъектов к некоторым критичным сущностям КС, несмотря на наличие необходимых прав доступа данных субъектов к этим сущностям [4].

Замечание 3. Существуют сущности, параметрически-ассоциированные с субъектом, которые не являются функционально-ассоциированными с ним. Примером такой сущности является раздел реестра ОС семейства Windows, содержащий информацию об установленных обновлениях ОС узла. Также существуют сущности, параметрически-ассоциированные с субъектом, которые являются функционально-ассоциированными с ним. Так, зная идентификатор сессии пользователя web-приложения, возможно получить его права доступа; с другой стороны, удаление данного идентификатора приводит к закрытию сессии пользователя.

Наличие у субъекта данных о параметрах функционирования другого субъекта КС может позволить получить первому субъекту право доступа владения ко второму субъекту. При этом первому субъекту необходимо иметь возможность реализовать информационный поток по памяти к некоторому субъекту, позволяющему получить права доступа ко второму субъекту. Например, при получении субъектом-нарушителем пароля доверенного субъекта первому необходимо иметь возможность записи пароля в сущность-интерфейс КС. С учетом того, что доверенные субъекты не участвуют в реализации информационных потоков по времени, и того, что в современных КС, как правило, реализация информационного потока по времени от сущности, параметрически-ассоциированной с субъектом, не приводит к получению другим субъектом права доступа владения к первому, будем считать, что в КС выполняется следующее предположение.

Предположение 1. Информационный поток по времени от сущности, параметрически-ассоциированной с субъектом, не приводит к получению другим субъ-

ектом права доступа владения к первому субъекту. Если субъект реализовал информационный поток по памяти от сущности, параметрически-ассоциированной с другим субъектом, к себе, то первый субъект получает право доступа владения ко второму субъекту. Множество сущностей, параметрически-ассоциированных с субъектом, не изменяется в процессе функционирования КС.

Через $]s[\subset E$ обозначим множество всех сущностей, параметрически-ассоциированных с субъектом s. При этом будем считать, что $s \in]s[$.

В соответствии с данным предположением модифицируем определение ФАС ДП-модели для возможности анализа условий получения субъектом права доступа владения к другому субъекту с использованием реализации информационного потока по памяти от сущности, параметрически-ассоциированной с последним субъектом. Модифицированную ФАС ДП-модель будем называть *ДП-моделью с функционально- и параметрически-ассоциированными с субъектами сущностями*, или ФПАС ДП-моделью. Модификация состоит в добавлении к правилам преобразования состояний ФАС ДП-модели правила $know(x, y, z)$, определенного в таблице: аргументом операции является состояние G, значением — состояние G', параметрами — сущности x, y, z.

Условия и результаты применения правила *know(x, y, z)* ФПАС ДП-модели

Правило	Исходное состояние $G = (S, E, R \cup A \cup F, H)$	Результирующее состояние $G' = (S', E', R' \cup A' \cup F', H')$
$know(x, y, z)$	$x, y \in S, z \in E, z \in]y[$ и или $x = z$, или $(z, x, write_m) \in F$	$S' = S, E' = E, A' = A, H' = H, F' = F, R' = R \cup (x, y, ownr)$

Замечание 4. Правило $know(x, y, z)$ является монотонным, то есть применение данного правила не приводит к удалению ребер или вершин из графа доступа.

Замечание 5. Как и правило $control(x, y, z)$ ФАС ДП-модели, правило $know(x, y, z)$ отражает возможность одним субъектом получить право доступа владения к другому субъекту путем реализации информационного потока.

Рассмотрим условия получения недоверенным субъектом права доступа владения own_r к другому субъекту без кооперации с ним для случая, когда в КС $\Sigma(G^*, OP)$ используются правила преобразования состояний $know(x, y, z)$ и $control(x, y, z)$.

2. Анализ условий получения субъектом права доступа владения к другому субъекту

Определение 2. Траекторию функционирования КС $\Sigma(G^*, OP)$ будем называть *траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа*, если при ее реализации используются монотонные правила преобразования состояний, и доверенные субъекты:

— не дают недоверенным субъектам права доступа к сущностям;
— не берут у недоверенных субъектов права доступа к сущностям;
— используя информационные потоки по памяти к сущностям, не получают право доступа владения к субъектам;
— используя информационные потоки по памяти от сущностей, не получают право доступа владения к субъектам.

Таким образом, в КС $\Sigma(G^*, OP)$ на траекториях без кооперации доверенных и недоверенных субъектов для передачи прав доступа доверенные субъекты:

— не инициируют выполнения следующих правил преобразования состояний: $take\_right(\alpha_r, u, x, e)$, $grant\_right(\alpha_r, u, x, e)$, $control(u, y, z)$, $know(u, y, z)$;
— доверенные субъекты могут выполнять монотонные правила преобразования состояний: $own\_take(\alpha_r, u, e)$, $create\_entity(u, e, e')$, $create\_subject(u, e, e')$, $rename\_entity(u, e, e')$, $access\_read(u, e)$, $access\_write(u, e)$, $access\_append(u, e)$, $find(u, e, e')$, $post(u, e, e')$, $pass(u, e, e')$ с условиями и результатами применения в соответствии с правилами преобразования БК ДП-модели,

где $u \in L_S$ — доверенный субъект, $x \in N_S$ — недоверенный субъект, y — субъект, что $z \in]y[$ или $z \in [y]$, z, e, e' — сущности, $\alpha_r \in R_r$ — право доступа.

Определение 3. Траекторию функционирования КС $\Sigma(G^*, OP)$ будем называть *траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных потоков*, если она является траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа, и при ее реализации используются правила преобразования состояний:

— $take\_right(\alpha_r, x, y, z)$, $grant\_right(\alpha_r, x, y, z)$, $own\_take(\alpha_r, x, y)$ с условиями и результатами применения в соответствии с правилами преобразования базовой ДП-модели;
— $create\_entity(x, y, z)$, $create\_subject(x, y, z)$, $rename\_entity(x, y, z)$, $flow(x, y, y', z)$, $find(x, y, z)$, $post(x, y, z)$, $pass(x, y, z)$, $access\_read(x, y)$, $access\_write(x, y)$, $access\_append(x, y)$ с условиями и результатами применения в соответствии с правилами преобразования БК ДП-модели;
— $control(x, y, z)$, $know(x, y, z)$ с условиями и результатами применения в соответствии с правилами преобразования ФАС ДП-модели и таблицей.

Определение 4. Пусть имеются $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние КС $\Sigma(G^*, OP)$, недоверенный субъект $x \in N_S \cap S_0$ и $y \in S_0$, где $x \neq y$. Определим предикат $can\_steal\_own(x, y, G_0, L_S)$, который будет истинным тогда и только тогда, когда 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ является траекторией без кооперации доверенных и недоверенных субъектов для передачи прав доступа и реализации информационных потоков и $(x, y, own_r) \in R_N$, где $N \geqslant 0$. При этом в последовательности правил $op_1, \ldots, op_N$ отсутствуют правила вида $grant\_right(\alpha_r, y, s, e)$, $take\_right(\alpha_r, y, s, e)$, $control(y, s, e')$, $know(y, s, e')$, где $\alpha_r \in R_r$, $s \in N_S \cap S_0$, $e, e' \in E_0$ и $e' \in [s]$ или $e' \in]s[$ (субъект y не передает другим субъектам любые имеющиеся у него права доступа к любым сущностям и не получает с использованием правил $control(x, y, z)$ и $know(x, y, z)$ право доступа владения own_r к другим субъектам).

Определение 5. Пусть имеются $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние КС $\Sigma(G^*, OP)$, недоверенный субъект $x \in N_S \cap S_0$ и $y \in S_0$, где $x \neq y$. Определим предикат $directly\_can\_share\_own(x, y, G_0, L_S)$, который будет истинным тогда и только тогда, когда существует последовательность субъектов $s_1, \ldots, s_m \in S_0$, где $s_1 = x$, $s_m =$ y и $m \geqslant 2$, таких, что для каждого $i = 1, ..., m-1$ выполняется одно из условий:

1) $s_i \in N_S \cap S_0$, $s_i \in [s_{i+1}]$ или $s_i \in]s_{i+1}[$;
2) истинен предикат $can\_share(own_r, s_i, s_{i+1}, G_0, L_S)$;
3) существует сущность $e \in [s_{i+1}]$, что предикат $can\_write\_memory(s_i, e, G_0, L_S)$ является истинным;

4) существует сущность $e \in]s_{i+1}[$, что предикат $can\_write\_memory(e, s_i, G_0, L_S)$ является истинным.

Лемма 1. Пусть имеются $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние КС $\Sigma(G^*, OP)$, недоверенный субъект $x \in N_S \cap S_0$ и $y \in S_0$, где $x \neq y$, и истинен предикат $directly\_can\_share\_own(x, y, G_0, L_S)$. Тогда истинен предикат $can\_share\_own(x, y, G_0, L_S)$.

Доказательство. Пусть истинен предикат $directly\_can\_share\_own(x, y, G_0, L_S)$, тогда по определению 5 существует последовательность субъектов $s_1, \ldots, s_m$ в S_0, где $s_1 = x$, $s_m = y$ и $m \geqslant 2$, таких, что для каждого $i = 1, ..., m-1$ выполняется одно из условий 1 – 4 определения 5. Докажем, что для такой последовательности $s_1, \ldots, s_m$ предикат $can\_share\_own(x, y, G_0, L_S)$ является истинным. Проведем доказательство этого утверждения индукцией по длине m.

Пусть $m = 2$, тогда возможно четыре случая.

Первый случай: $x \in N_S \cap S_0$, $x \in [y]$ или $x \in]y[$. Если $x \in [y]$, то пусть $op_1 = control(x, y, x)$. Тогда $G_0 \vdash_{op1} G_1$, $(x, y, own_r) \in R_1$ и предикат $can\_share\_own(x, y, G_0, L_S)$ истинен. Если $x \in]y[$, то пусть $op_1 = know(x, y, x)$. Тогда $G_0 \vdash_{op1} G_1$ и $(x, y, own_r) \in R_1$ и предикат $can\_share\_own(x, y, G_0, L_S)$ также истинен.

Во втором случае истинен предикат $can\_share(x, y, G_0, L_S)$. Следовательно, истинен предикат $can\_share\_own(x, y, G_0, L_S)$.

В третьем случае имеется $x \in N_S \cap S_0$, существует сущность $e \in [y]$ и истинен предикат $can\_write\_memory(x, e, G_0, L_S)$. Пусть $op_1 = control(x, y, e)$, тогда $G_0 \vdash_{op1} G_1$, $(x, y, own_r) \in R_1$ и предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

В четвертом случае имеется $x \in N_S \cap S_0$, существует сущность $e \in]y[$ и истинен предикат $can\_write\_memory(e, x, G_0, L_S)$. Пусть $op_1 = know(x, y, e)$, тогда $G_0 \vdash_{op1} G_1$, $(x, y, own_r) \in R_1$ и предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

Докажем индуктивный шаг. Пусть $m > 2$ и доказываемое утверждение верно для всех последовательностей субъектов длины $k < m$. Докажем, что оно верно и для всех таких последовательностей длины m.

Рассмотрим последовательность субъектов $s_1, \ldots, s_m$ в S_0, где $s_1 = x$, $s_2 = z$ и $s_m = y$. Пусть $z \in N_S \cap S_0$. Тогда по предположению индукции существуют правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$, и верно, что (x, z, own_r), $(z, y, own_r) \in R_N$, где $N \geqslant 0$. Положим $op_{N+1} = take\_right(own_r, x, z, y)$. Тогда $G_N \vdash_{op(N+1)} G_{N+1} = (S_{N+1}, E_{N+1}, R_{N+1} \cup A_{N+1} \cup F_{N+1}, H_{N+1})$ и $(x, y, own_r) \in R_{N+1}$, следовательно, предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

Если $z \in L_S \cap S_0$, то $(z, y, own_r) \in R_0$, и по предположению индукции предикат $can\_share\_own(x, z, G_0, L_S)$ истинен. Следовательно, существуют правила преобразования состояний $op_1, \ldots, op_N$, такие, что $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$, и $(x, z, own_r) \in R_N$, где $N \geqslant 0$. Аналогично получаем истинность предиката $can\_share\_own(x, y, G_0, L_S)$. Лемма доказана. ■

Определим и обоснуем алгоритмически проверяемые необходимые и достаточные условия истинности предиката $can\_steal\_own(x, y, G_0, L_S)$.

Теорема 1. Пусть имеется $G_0 = (S_0, E_0, R_0 \cup A_0 \cup F_0, H_0)$ — состояние КС $\Sigma(G^*, OP)$, и $(x, y, own_r) \in N_r$, где $x \in N_S \cap S_0$, $y \in S_0$ и $x \neq y$. Тогда предикат $can\_steal\_own(x, y, G_0, L_S)$ является истинным, если и только если существует по-

следовательность субъектов $s_1, \ldots, s_m$ в S_0, где $s_1 = x$, $s_m = y$ и $m \geqslant 2$, таких, что выполняется одно из условий.

Условие 1. $m = 2$ и истинен предикат $directly\_can\_share\_own(x, y, G_0, L_S)$.

Условие 2. $m > 2$ и для каждого $i = 1, ..., m-2$ выполняется одно из условий:

— $s_i, s_{i+1} \in N_S \cap S_0$, $s_i \neq y$ и предикаты $directly\_can\_share\_own(s_i, s_{i+1}, G_0, L_S)$, $directly\_can\_share\_own(s_{i+1}, s_{i+2}, G_0, L_S)$ являются истинными;

— $i < m-2$, $s_i, s_{i+2} \in N_S \cap S_0$, $s_i, s_{i+2} \neq y$ и являются истинными предикаты $directly\_can\_share\_own(s_i, s_{i+1}, G_0, L_S)$ и $directly\_can\_share\_own(s_{i+2}, s_{i+1}, G_0, L_S)$;

— $i < m-2$, $s_{i+1}, s_{i+2} \in N_S \cap S_0$, $s_{i+1} \neq y$, $s_{i+2} \neq y$ и являются истинными предикаты $directly\_can\_share\_own(s_{i+1}, s_i, G_0, L_S)$, $directly\_can\_share\_own(s_{i+2}, s_{i+1}, G_0, L_S)$;

— $s_{i+1} \in N_S \cap S_0$, $s_{i+1} \neq y$ и являются истинными предикаты $directly\_can\_share\_own(s_{i+1}, s_i, G_0, L_S)$, $directly\_can\_share\_own(s_{i+1}, s_{i+2}, G_0, L_S)$.

Доказательство. Докажем достаточность выполнения условий теоремы для истинности предиката $can\_steal\_own(x, y, G_0, L_S)$. Если выполнено первое условие теоремы, то в соответствии с леммой истинен предикат $can\_share\_own(x, y, G_0, L_S)$. Следовательно, предикат $can\_steal\_own(x, y, G_0, L_S)$ также является истинным. Если выполнено второе условие теоремы, то тогда существует последовательность субъектов $s_1, ..., s_m$ в S_0, где $s_1 = x$, $s_m = y$, $x \neq y$ и $m > 2$.

Проведем доказательство индукцией по длине m последовательности субъектов. Пусть $m = 3$. Возможны два случая. Первый случай: $x, s_2 \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(x, s_2, G_0, L_S)$, $directly\_can\_share\_own(s_2, y, G_0, L_S)$. Тогда в соответствии с леммой 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и (x, s_2, own_r), $(s_2, y, own_r) \in R_N$, где $N \geqslant 0$. Пусть $op_{N+1} = take\_right(own_r, x, s_2, y)$ и $G_N \vdash_{op(N+1)} G_{N+1}$, где $G_{N+1} = (S_{N+1}, E_{N+1}, R_{N+1} \cup A_{N+1} \cup F_{N+1}, H_{N+1})$. Тогда $(x, y, own_r) \in R_{N+1}$ и предикат $can\_steal\_own(x, y, G_0, L_S)$ является истинным. Второй случай: $s_2 \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(s_2, x, G_0, L_S)$, $directly\_can\_share\_own(s_2, y, G_0, L_S)$. Тогда в соответствии с леммой 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и (s_2, x, own_r), $(s_2, y, own_r) \in R_N$, где $N \geqslant 0$. Пусть $op_{N+1} = grant\_right(own_r, s_2, x, y)$ и $G_N \vdash_{op(N+1)} G_{N+1}$, где $G_{N+1} = (S_{N+1}, E_{N+1}, R_{N+1} \cup A_{N+1} \cup F_{N+1}, H_{N+1})$. Тогда $(x, y, own_r) \in R_{N+1}$ и предикат $can\_steal\_own(x, y, G_0, L_S)$ является истинным.

Докажем индуктивный шаг. Пусть $m > 4$ и утверждение теоремы верно для всех последовательностей субъектов длины $k < m$. Докажем, что утверждение теоремы верно для всех таких последовательностей длины m. Пусть имеется последовательность субъектов $s_1, \ldots, s_m$ в S_0, где $s_1 = x$, $s_2 = z$, $s_3 = w$ и $s_m = y$. Возможно четыре случая.

Первый случай: $x, z \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(x, z, G_0, L_S)$, $directly\_can\_share\_own(z, w, G_0, L_S)$. Тогда в соответствии с леммой 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и $(x, z, own_r), (z, w, own_r) \in R_N$, где $N \geqslant 0$. По предположению индукции существуют состояния $G_{N+1}, \ldots, G_{N+K} = (S_{N+K}, E_{N+K}, R_{N+K} \cup A_{N+K} \cup F_{N+K}, H_{N+K})$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, такие, что $G_{N+1} \vdash_{op(N+1)} G_{N+2} \vdash_{op(N+2)}$

$\ldots \vdash_{op(N+K)} G_{N+K}$ и $(w, y, own_r) \in R_{N+K}$, где $K \geqslant 0$. Положим $op_{N+K+1} = take\_right(own_r, x, z, w)$, $op_{N+K+2} = take\_right(own_r, x, w, y)$, тогда $G_{N+K} \vdash_{op(N+K+1)} G_{N+K+1} \vdash_{op(N+K+2)} G_{N+K+2} = (S_{N+K+2}, E_{N+K+2}, R_{N+K+2} \cup A_{N+K+2} \cup F_{N+K+2}, H_{N+K+2})$ и $(x, y, own_r) \in R_{N+K+2}$. Следовательно, предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

Второй случай: $x, w \in N_S \cap S_0$ и истинны предикаты $directly\_can\_share\_own(x, z, G_0, L_S)$, $directly\_can\_share\_own(w, z, G_0, L_S)$. Тогда в соответствии с леммой 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и $(x, z, own_r), (w, z, own_r) \in R_N$, где $N \geqslant 0$. По предположению индукции существуют состояния $G_{N+1}, \ldots, G_{N+K} = (S_{N+K}, E_{N+K}, R_{N+K} \cup A_{N+K} \cup F_{N+K}, H_{N+K})$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, такие, что $G_{N+1} \vdash_{op(N+1)} G_{N+2} \vdash_{op(N+2)} \ldots \vdash_{op(N+K)} G_{N+K}$ и $(w, y, own_r) \in R_{N+K}$, где $K \geqslant 0$. Положим $op_{N+K+1} = grant\_right(own_r, w, z, y)$, $op_{N+K+2} = take\_right(own_r, x, z, y)$, тогда $G_{N+K} \vdash_{op(N+K+1)} G_{N+K+1} \vdash_{op(N+K+2)} G_{N+K+2} = (S_{N+K+2}, E_{N+K+2}, R_{N+K+2} \cup A_{N+K+2} \cup F_{N+K+2}, H_{N+K+2})$ и $(x, y, own_r) \in R_{N+K+2}$. Следовательно, предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

Третий случай: $w, z \in N_S \cap S_0$, $z \neq y$ и истинны предикаты $directly\_can\_share\_own(z, x, G_0, L_S)$, $directly\_can\_share\_own(w, z, G_0, L_S)$. Тогда в соответствии с леммой 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и $(z, x, own_r), (w, z, own_r) \in R_N$, где $N \geqslant 0$. По предположению индукции существуют состояния $G_{N+1}, \ldots, G_{N+K} = (S_{N+K}, E_{N+K}, R_{N+K} \cup A_{N+K} \cup F_{N+K}, H_{N+K})$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, такие, что $G_{N+1} \vdash_{op(N+1)} G_{N+2} \vdash_{op(N+2)} \ldots \vdash_{op(N+K)} G_{N+K}$ и $(w, y, own_r) \in R_{N+K}$, где $K \geqslant 0$. Положим $op_{N+K+1} = grant\_right(own_r, w, z, y)$, $op_{N+K+2} = grant\_right(own_r, z, x, y)$, тогда $G_{N+K} \vdash_{op(N+K+1)} G_{N+K+1} \vdash_{op(N+K+2)} G_{N+K+2} = (S_{N+K+2}, E_{N+K+2}, R_{N+K+2} \cup A_{N+K+2} \cup F_{N+K+2}, H_{N+K+2})$ и $(x, y, own_r) \in R_{N+K+2}$. Следовательно, предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

Четвертый случай: $z \in N_S \cap S_0$, $z \neq y$ и истинны предикаты $directly\_can\_share\_own(z, x, G_0, L_S)$, $directly\_can\_share\_own(z, w, G_0, L_S)$. Тогда в соответствии с леммой 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$ и $(z, x, own_r), (z, w, own_r) \in R_N$, где $N \geqslant 0$. По предположению индукции существуют состояния $G_{N+1}, \ldots, G_{N+K} = (S_{N+K}, E_{N+K}, R_{N+K} \cup A_{N+K} \cup F_{N+K}, H_{N+K})$ и правила преобразования состояний $op_{N+1}, \ldots, op_{N+K}$, такие, что $G_{N+1} \vdash_{op(N+1)} G_{N+2} \vdash_{op(N+2)} \ldots \vdash_{op(N+K)} G_{N+K}$ и $(w, y, own_r) \in R_{N+K}$, где $K \geqslant 0$. Положим $op_{N+K+1} = take\_right(own_r, z, w, y)$, $op_{N+K+2} = grant\_right(own_r, z, x, y)$, тогда $G_{N+K} \vdash_{op(N+K+1)} G_{N+K+1} \vdash_{op(N+K+2)} G_{N+K+2} = (S_{N+K+2}, E_{N+K+2}, R_{N+K+2} \cup A_{N+K+2} \cup F_{N+K+2}, H_{N+K+2})$ и $(x, y, own_r) \in R_{N+K+2}$. Следовательно, предикат $can\_share\_own(x, y, G_0, L_S)$ истинен.

Индуктивный шаг доказан. Доказательство достаточности выполнения условий теоремы для истинности предиката $can\_steal\_own(x, y, G_0, L_S)$ закончено.

Докажем необходимость выполнения условий теоремы для истинности предиката $can\_steal\_own(x, y, G_0, L_S)$.

Пусть истинен предикат $can\_steal\_own(x, y, G_0, L_S)$. Тогда по определению 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 $G_0 \vdash_{op1} G_1 \vdash_{op2} \ldots \vdash_{opN} G_N$, и $(x, y, own_r) \in R_N$,

где $N \geqslant 0$. Выберем среди последовательностей правил преобразований ту, у которой длина N является минимальной. Следовательно, $(x, y, own_r) \notin R_{N-1}$. При этом в последовательности правил $op_1, \ldots, op_N$ отсутствуют правила вида $grant\_right(\alpha, y, s, e)$, $take\_right(\alpha, y, s, e)$, $control(y, s, e')$, $know(y, s, e')$, где $alpha \in R_r$, $s \in N_S \cap S_0$, $e, e' \in E_0$ и $e' \in [s]$ или $e' \in]s[$. Проведем доказательство индукцией по числу N.

Пусть $N = 0$, тогда $(x, y, own_r) \in R_0$ и условие 1 теоремы выполнено. Пусть $N = 1$, тогда $x \in N_S \cap S_0$, $y \in S0$, $(x, y, own_r) \notin R_0$ и существует правило преобразования состояний op_1, такое, что $G_0 \vdash_{op1} G_1$ и $(x, y, own_r) \in R_1$. Из определения правил преобразования состояний следует, что возможны шесть случаев:

— $x \in [y]$ и $op_1 = control(x, y, x)$;
— $x \in]y[$ и $op_1 = know(x, y, x)$;
— существует сущность $e \in [y]$, такая, что $(x, e, write_m) \in F_0$ и $op_1 = control(x, y, e)$;
— существует сущность $e \in]y[$, такая, что $(e, x, write_m) \in F_0$ и $op_1 = know(x, y, e)$;
— существует субъект $z \in N_S \cap S_0$, такой, что (x, z, own_r), $(x, z, own_r) \in R_0$ и $op_1 = take\_right(own_r, x, z, y)$;
— существует субъект $z \in N_S \cap S_0$, такой, что $(z, x, own_r), (x, z, own_r) \in R_0$ и $op_1 = grant\_right(own_r, z, x, y)$.

Все шесть случаев соответствуют условиям 1 и 2 теоремы.

Пусть $N > 2$ и утверждение теоремы верно для всех последовательностей преобразований состояний длины $l < N$. Тогда $x \in N_S \cap S_0$, $y \in S_0$, $(x, y, own_r) \notin R_{N-1}$ и существует правило преобразования состояний op_N, такое, что $G_{N-1} \vdash_{opN} G_N$ и $(x, y, own_r) \in R_N$.

Из определения правил преобразования состояний и минимальности N следует, что выполняется одно из условий:

— существует сущность $e \in [y]$, такая, что $(x, e, write_m) \in F_{N-1}$ и $op_N = control(x, y, e)$;
— существует сущность $e \in]y[$, такая, что $(e, x, write_m) \in F_{N-1}$ и $op_N = know(x, y, e)$;
— существует субъект $z \in N_S \cap S_0$, такой, что $(x, z, own_r), (z, y, own_r) \in R_{N-1}$ и $op_N = take\_right(own_r, x, z, y)$;
— существует субъект $z \in N_S \cap S_0$, такой, что $(z, x, own_r), (z, y, own_r) \in R_{N-1}$ и $op_N = grant\_right(own_r, z, x, y)$.

Если выполнено первое или второе условие, то выполнено первое условие теоремы. Рассмотрим случай выполнения третьего условия, когда $(x, z, own_r), (z, y, own_r) \in R_{N-1}$ и $op_N = take\_right(\alpha, x, z, y)$. Доказательство для случая выполнения четвертого условия проводится аналогично доказательству для случая выполнения третьего условия.

Так как длина N минимальна, то в последовательности преобразований состояний не использовались правила вида $create\_entity(x, y, z)$ и $create\_subject(x, y, z)$. Следовательно, $z \in S_0$ и истинны предикаты $can\_steal\_own(x, z, G_0, L_S)$ и $can\_steal\_own(z, y, G_0, L_S)$ с длинами последовательностей преобразований состояний меньше N. По предположению индукции возможны четыре случая. Первый случай: истинны предикаты $directly\_can\_share\_own(x, z, G_0, L_S)$ и $directly\_can\_share\_own(z, y, G_0, L_S)$. Следовательно, второе условие теоремы выполнено.

Второй случай: истинен предикат $directly\_can\_share\_own(x, z, G_0, L_S)$ и существует последовательность субъектов $s_1, \ldots, s_m$ в S_0, где $s_1 = z, s_m = y$ и $m \geqslant 2$, таких, что выполняется условие 2. Следовательно, существует последовательность субъектов $s_1, \ldots, s_m, s_{m+1}$ в S_0, где $s_1 = x$, $s_2 = z$, $s_{m+1} = y$, таких, что выполняется условие 2 теоремы.

Третий случай: истинен предикат $directly\_can\_share\_own(z, y, G_0, L_S)$ и существует последовательность субъектов $s_1, \ldots, s_m$ в S_0, где $s_1 = x, s_m = z$ и $m \geqslant 2$, таких, что выполняется условие 2. Следовательно, существует последовательность субъектов $s_1, \ldots, s_m, s_{m+1}$ в S_0, где $s_1 = x$, $s_m = z$, $s_{m+1} = y$, таких, что выполняется условие 2 теоремы.

Четвертый случай: существуют последовательности субъектов $s_1, \ldots, s_m$ и $s'_1, \ldots, s'_n$ в S_0, где $s_1 = x, s_m = s'_1 = z$, $s'_n = y$ и $m, n \geqslant 2$, для которых выполняется условие 2. Следовательно, существует последовательность субъектов $s''_1, \ldots, s''_{m+n-1}$ в S_0, где $s''_1 = x$, $s''_{m+n-1} = y$, таких, что выполняется условие 2 теоремы.

Индуктивный шаг доказан. Доказательство необходимости выполнения условия теоремы для истинности предиката $can\_steal\_own(x, y, G_0, L_S)$ закончено. Теорема доказана. ■

ЛИТЕРАТУРА

1. *Девянин П. Н.* Анализ безопасности управления доступом и информационными потоками в компьютерных системах. М.: Радио и связь, 2006. 176 с.
2. *Девянин П. Н.* Модели безопасности компьютерных систем: Учеб. пособие для студ. высш. учеб. заведений. М.: Издательский центр «Академия», 2005. 144 с.
3. *Робачевский А.* Операционная система UNIX. СПб.: БХВ-Петербург, 2000. 528 с.
4. *Качанов М. А., Колегов Д. Н.* Расширение функциональности системы безопасности ядра Linux на основе подмены системных вызовов // Прикладная дискретная математика. 2008. № 2. С. 76 – 80.

СВЕДЕНИЯ ОБ АВТОРАХ

БУРЕНИН Павел Валерьевич — заместитель директора ООО «Твест». E-mail: **troy1f4@mail.ru**

ДУЛЬКЕЙТ Владимир Игоревич — аспирант, Омский государственный университет. E-mail: **vidulkeyt@mail.ru**

ЗАКРЕВСКИЙ Аркадий Дмитриевич — член-корр. НАН Беларуси, доктор технических наук, главный научный сотрудник Объединённого института проблем информатики НАН Беларуси, г. Минск. E-mail: **zakrevskij@tut.by**

КОЛЕГОВ Денис Николаевич — аспирант Томского государственного университета. E-mail: **d.n.kolegov@gmail.com**

ПРОКОПЬЕВ Сергей Евгеньевич — независимый исследователь. E-mail: **prsz@bk.ru**

ТОКАРЕВА Наталья Николаевна — кандидат физико-математических наук, научный сотрудник, Институт математики им. С. Л. Соболева СО РАН, г. Новосибирск. E-mail: **tokareva@math.nsc.ru**

ТОРОПОВ Николай Романович — кандидат физико-математических наук, ведущий научный сотрудник Объединённого института проблем информатики НАН Беларуси, г. Минск.

ФАЙЗУЛЛИН Рашит Тагирович — профессор, доктор технических наук, Омский государственный технический университет. E-mail: **r.t.faizullin@mail.ru**

ШОЛОМОВ Лев Абрамович — профессор, доктор физико-математических наук, ведущий научный сотрудник, Институт системного анализа РАН, г. Москва. E-mail: **sholomov@isa.ru**

АННОТАЦИИ СТАТЕЙ НА АНГЛИЙСКОМ ЯЗЫКЕ

Zakrevskij A. D., Toropov N. R. **MINIMIZATION OF BOOLEAN FUNCTIONS OF MANY VARIABLES — ITERATIVE METHOD AND PROGRAM REALIZATION.** An iterative method for minimization of Boolean functions depending on the large number n (up to 25) of variables is proposed. The method is based on applying effective parallel operations on Boolean vectors of length 2^n.

Tokareva N. N. **BENT FUNCTIONS: RESULTS AND APPLICATIONS. A SURVEY.** A survey of main results on bent functions is given. Theoretical and practical applications of bent functions are considered.

Sholomov L. A. **LOGICAL METHODS FOR DESIGN AND ANALYSIS OF CHOICE MODELS.** The logical methods use representations of choice functions and choice models by means of formulas of some logic language. The design and research problems for choice models are reduced to formal transformations and analysis of the presentations. The logical methods make it possible to solve a wide range of constructive problems associated with a design, an analysis, simplifications, and estimations of complexity for formal choice models. They allow to use to choice models the Shannon—Yablonsky—Lupanov methodology developed for investigation of computing systems. The article systematizes obtained at different times and published in different editions the author's results on a study of choice models by logical methods. Results of other authors concerning the topics are brought, also.

Dulkeyt V. I., Faizulin R. T. **APPROXIMATE SOLUTION OF THE TRAVELING SALESMAN PROBLEM.** A heuristic algorithm for the approximate solution of traveling salesman problem is proposed in the article.

Prokopyev S. E. **MODELLING OF THE PKI PROTOCOLS IN THE UNIVERSALLY COMPOSABLE FRAMEWORK USING MODEL CHECKERS.** We analyze the PKI protocols in the universally composable security framework with following purposes: 1) decomposition of the code of the cryptographic service "Digital Signature with PKI" to the high- and low-danger parts, 2) obtaining a deterministic and cryptographically sound abstraction of this service. We experimented with NuSMV model checker to automate partially our analysis.

Burenin P. V. **APPROACHES TO THE CONSTRUCTION OF THE DP-MODEL OF FILE SYSTEMS.** Approaches to the creation of the DP-model of file systems are listed in the article. DP-model family of computer systems with discretionary management of access is the basis of this approach. Specific conditions of subject's functioning, rights transfer conditions and realization of information streams are considered in the article on the basis of DP-model. Also sufficient conditions for realization of forbidden information memory streams in file systems are proved.

Kolegov D. N. **USAGE FORMAL MODELS FOR VULNERABILITY ANALYSIS.** In the paper the formal approach to vulnerability analysis based on mathematical security models of the computer systems is considered. The attacker model proposed in "Security assessment information technology criteria" is constructed and mathematical definition of the penetration stability is proposed in the terms of the DP-model.

Kolegov D. N. **SECURITY ANALYSIS OF THE INFORMATION FLOWS BY MEMORY IN THE COMPUTER SYSTEMS WITH FUNCTIONAL AND PARAMETRIC ASSOCIATED ENTITIES.** The definition of the parametric associated entities with computer systems subjects is proposed. The DP-model extension included this entities is built.