Асимптотически оптимальный алгоритм расшифровки разбиения булевого куба на подкубы

В. В. Осокин

Задача расшифровки функций алгебры логики, то есть задача восстановления значений функции на всех наборах n-мерного булева куба по известным значениям на некоторых из этих наборов, решалась для конкретных классов функций, таких, например, как монотонные, пороговые. Практический интерес представляет рассмотрение задачи для класса функций, задающих разбиение n-мерного булева куба на подкубы. Алгоритмы расшифровки таких функций могут быть применены в важной с прикладной точки зрения задаче классификации документов по тематической принадлежности. Ранее автором был построен алгоритм расшифровки функций из любого подкласса исследуемого класса с фиксированной структурой. В настоящей работе удалось существенно расширить класс функций, поддающихся расшифровке. Доказана асимптотическая оптимальность построенного в работе алгоритма.

1. Введение.

В данной работе рассматривается задача расшифровки функций, определенных на n-мерном булевом кубе и принимающих значения из $\mathbb{N}$, в ее стандартной постановке. В общем случае для однозначного определения функции, заданной на n-мерном булевом кубе, требуется знать ее значения на всех 2^n наборах куба. Однако, если рассматривать более узкий класс функций, чем класс всех функций от n булевых переменных, то может потребоваться меньшее число наборов. А именно, может найтись алгоритм, который за меньшее чем

2^n число обращений к «черному ящику» восстанавливает таблицу значений любой функции из заданного класса (под «черным ящиком» здесь подразумевается некоторый оператор, который «знает» искомую функцию, а под обращением к «черному ящику» — запрос значения упомянутого оператора на выбранном алгоритмом наборе).

Естественной мерой сложности алгоритмов расшифровки служит число наборов, запрашиваемых алгоритмом у «черного ящика» в худшем случае, то есть, для расшифровки «самой сложной» функции.

Впервые задача расшифровки функций в указанной постановке рассматривалась В. К. Коробковым для класса монотонных булевых функций [1]. Окончательные результаты, касающиеся сложности расшифровки функций из этого класса, получил Ж.Ансель [2]. Другим изученным классом функций является класс пороговых функций. Задача расшифровки пороговых функций поставлена и решалась В. Н. Шевченко [3], в данном направлении результаты также получены Н. Ю. Золотых [4]. Здесь необходимо упомянуть и область, близкую к рассматриваемой — теорию тестов. Последняя активно разрабатывается, в частности, в Московском [5] и Нижегородском [6] государственных университетах.

Аналогичная задача возникает и в так называемом «машинном обучении». Здесь наиболее широко исследованы два подхода. Первый в середине 80-х предложила Д. Англуин [7]. Он известен под названием «точное обучение» (exact learning) и достаточно просто сводится к задаче расшифровки функций в указанной ранее постановке. Второй подход носит название «вероятностно-аппроксимационно корректирующее обучение» (Probably Approximately Correct learning) и был предложен также в 80-х годах XX века (Л. Валиант, [8]). Основное его отличие заключается в том, что алгоритм в процессе обучения более не может сам выбирать наборы булева куба, подаваемые «черному ящику». Вместо этого при поступлении запроса «черный ящик», руководствуясь некоторым заранее заданным распределением вероятности на булевом кубе, сам выбирает набор n-мерного куба и выдает алгоритму пару (набор, значение функции на этом наборе).

В представленной работе нас будет интересовать класс функций, задающих разбиение n-мерного булева куба на подкубы. Обозначим этот класс через Ф. Каждая функция из этого класса некоторым

образом «разбивает» n-мерный куб на непересекающиеся подкубы, сопоставляя каждому такому подкубу его номер. Задача расшифровки таких функций впервые была поставлена в [9, 10] и решена для каждого подкласса Φ_R класса Φ, для которого заранее задана «структура» R этого подкласса. В настоящей работе построен алгоритм расшифровки функций из $\Phi_{\mathcal{R}} = \cup_{R\in\mathcal{R}}\Phi_R$ для любого конечного множества $\mathcal{R}$ и доказана асимптотическая неулучшаемость этого алгоритма в случае, когда с ростом n мощность $\mathcal{R}$ растет не быстрее чем $o(\log_2 n)$.

Перед тем, как перейти к строгой математической постановке задачи, приведем пример, когда и в каких целях может быть использована расшифровка функций указанного вида. Предположим, есть интернет-сервер, хранящий корпус документов. Пусть n — суммарное число слов по всем документам. Каждый документ задается набором n-мерного булева куба, на i-м месте которого стоит 1, если i-е слово присутствует в документе, и 0 в противном случае. Каждому набору-документу сопоставлено (с помощью некоторой функции $f : B^n \to \mathbb{N}$) натуральное число — номер темы в которой он лежит. Пусть теперь есть локальный сервер, который должен позволять локальному пользователю узнавать тему, в которой лежит поданный пользователем документ, без выхода в интернет. Перед тем, как начать работу с пользователем, локальный сервер должен полностью определить функцию f, запросив, предпочтительно, как можно меньшее число документов у интернет-сервера (например, в силу дороговизны трафика), то есть, расшифровать функцию f за минимальное число обращений к «черному ящику».

Результаты данной работы анонсированы в [11].

Автор выражает благодарность профессору Э. Э. Гасанову за постановку задачи и помощь в работе.

2. Постановка задачи и формулировка результатов

Рассмотрим всюдуопределенную на k-мерном булевом кубе B^k функцию $R : B^k \to \mathbb{N}$, такую что для любого числа i из области

$R(B^k)$ ее значений прообраз $R^{-1}(i)$ этого числа является гранью B^k. Всякую такую функцию R будем называть *функцией граневого разбиения*, руководствуясь тем, что эта функция разбивает k-мерный куб на непересекающиеся подкубы (грани) путем сопоставления уникального номера каждому подкубу разбиения.

Фиксируем некоторое $n \in \mathbb{N}$. Пусть $\{y_1, \ldots, y_n\}$ — булевы переменные, R — функция граневого разбиения арности k. Рассмотрим множество G_n^k всех функций $g:\ \{1, \ldots, k\} \to \{y_1, \ldots, y_n, 0, 1\}$. Положим

$$\Phi_R^n = \{R(g(1), \ldots, g(k)),\ g \in G_n^k\}$$

— это некоторый класс булевых функций от переменных $y_1, \ldots, y_n$, существенно зависящих не более чем от k переменных. Каждая функция из Φ_R^n разбивает n-мерный булев куб на не более чем 2^k непересекающихся подкубов.

Фиксируем множество $\mathcal{R} = \{R_1, \ldots, R_m\}$ различных функций граневого разбиения. Через $k(R_i)$, $i \in \{1, \ldots, m\}$, обозначим арность функции R_i. По определению положим

$$\Phi_{\mathcal{R}}^n = \Phi_{R_1}^n \cup \Phi_{R_2}^n \cup \ldots \cup \Phi_{R_m}^n$$

— класс функций, производящих разбиение n-мерного булева куба на не более чем $2^{\max\limits_{i \in \{1,\ldots,m\}} k(R_i)}$ непересекающихся подкубов.

Рассмотрим в качестве примера три функции граневого разбиения $R_1 : B^4 \to \{1, 2, 3, 4, 5\}$, $R_2 : B^3 \to \{1, 2, 3, 4\}$ и $R_3 : B^3 \to \{1, 2, 3, 4, 5\}$. Значения, которые данные функции принимают на наборах соответствующих булевых кубов, представлены на рис. 1.

Пусть $n = 5$, тогда $\Phi_{\mathcal{R}}^5 = \Phi_{R_1}^5 \cup \Phi_{R_2}^5 \cup \Phi_{R_3}^5$ — некоторое множество функций от пяти переменных y_1, y_2, y_3, y_4, y_5, и функция $f = R_1(y_1, y_3, y_5, 0)$ является примером функции из $\Phi_{\mathcal{R}}^5$.

$$f(y_1, \ldots, y_5) = \begin{cases} 2 & \text{при } y_1 = y_5 = 0,\ y_3 = 1; \\ 3 & \text{при } y_1 = y_3 = y_5 = 0; \\ 4 & \text{при } y_5 = 1; \\ 5 & \text{при } y_1 = 1,\ y_5 = 0. \end{cases} \tag{1}$$

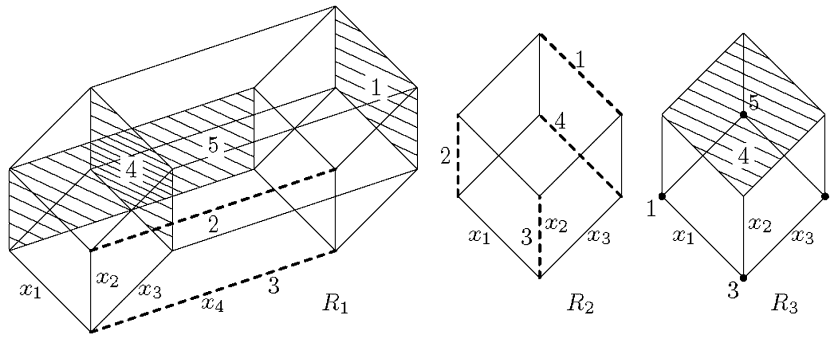

Рис. 1. Функции граневого разбиения R_1, R_2 и R_3.

В данной работе нас будет интересовать задача расшифровки функций из $\Phi^n_{\mathcal{R}}$ в ее стандартной постановке: будем считать, что задан оператор $\mathcal{A}_f$, вычисляющий для произвольного набора из B^n значение функции $f \in \Phi^n_{\mathcal{R}}$ на этом наборе. Требуется за минимальное число обращений к оператору $\mathcal{A}_f$ полностью восстановить таблицу значений функции $f(y_1, \ldots, y_n)$.

Фиксируем натуральные числа n, m и k. Обозначим через $\mathbf{R}(m,k)$ множество таких множеств $\mathcal{R} = \{R_1, \ldots, R_l\}$ функций граневого разбиения, что $l \leqslant m$ и $\max\limits_{i \in \{1,\ldots,l\}} k(R_i) \leqslant k$. Рассмотрим множество $\mathcal{F}$ алгоритмов, решающих указанную задачу для любого множества $\mathcal{R}$ из $\mathbf{R}(m,k)$. На вход любого такого алгоритма подаются функции $R_1, \ldots, R_l$ множества $\mathcal{R}$ и оператор $\mathcal{A}_f$, где $f \in \Phi^n_{\mathcal{R}}$. Работа алгоритма $F \in \mathcal{F}$ заключается в том, что он последовательно запрашивает значения оператора $\mathcal{A}_f$ на наборах из B^n. При этом алгоритм F предполагается условным, то есть при выборе очередного набора он может пользоваться знаниями о значениях функции на ранее поданных им наборах. Пусть $\varphi(F, \mathcal{R}, f)$ — число обращений к оператору $\mathcal{A}_f$ в процессе восстановления таблицы значений функции f с помощью алгоритма F. Обозначим

$$\varphi(n,m,k) = \min_{F \in \mathcal{F}} \max_{\mathcal{R} \in \mathbf{R}(m,k)} \max_{f \in \Phi^n_{\mathcal{R}}} \varphi(F, \mathcal{R}, f)$$

— сложность расшифровки самой плохой функции самым хорошим

алгоритмом. Целью настоящей работы является описание асимптотического поведения функции $\varphi(n, m, k)$ при $n \to \infty$. В работе доказана следующая теорема:

Теорема 1. *Если* $k \to \infty$ *при* $n \to \infty$ *и* $k \leqslant cn$, *где* $c < 1$, $m = o(\log_2 n)$, *то при* $n \to \infty$ *имеет место*

$$\varphi(n, m, k) \sim k \log_2 n.$$

3. Нижняя оценка

Согласно доказанной в [10] нижней оценке имеем $\varphi(n, 1, k) \geqslant (k -] \log_2 k[)] \log_2(n - k + 1)[$. Легко видеть, что для любых m, n и k выполнено $\varphi(n, m, k) \geqslant \varphi(n, 1, k)$. Тем самым справедлива следующая лемма.

Лемма 1. *Для любых натуральных* n, m, k *имеет место*

$$\varphi(n, m, k) \geqslant (k -] \log_2 k[)] \log_2(n - k + 1)[.$$

4. Верхняя оценка

Согласно [10] упорядоченное множество наборов A из B^n назовем *правильным*, если в матрице, строками которой являются эти наборы, все n столбцов различны. Эту матрицу также будем обозначать через A. Очевидно, что минимальное по мощности правильное множество состоит из $] \log_2 n[$ элементов.

Для произвольной матрицы A размера $q \times n$ с элементами из $\{0, 1\}$ и набора $a \in B^n$ обозначим через $A \cdot a$ ($A \vee a$) матрицу $q \times n$, такую что $(A \cdot a)_{ij} = A_{ij} \cdot a_i$ (соответственно $(A \vee a)_{ij} = A_{ij} \vee a_i$). Для произвольной матрицы A размера $q \times n$ с элементами из $\{0, 1\}$ и набора $a \in \{0, 1, 2\}^n$ обозначим через $A \diamond a$ матрицу $q \times n$, такую что

$$(A \diamond a)_{ij} = \begin{cases} a_i, \text{ если } a_i \neq 2; \\ A_{ij} \text{ в противном случае.} \end{cases}$$

Пусть B^k — k-мерный булев куб от переменных $x_1, \ldots, x_k$. Пусть $B = \{(\alpha_1, \ldots, \alpha_k) \in B^k : \ \alpha_{i_1} = \sigma_1, \ldots, \alpha_{i_s} = \sigma_s\}$ — некоторый его подкуб. Будем говорить, что подкуб B задается элементарной конъюнкцией $K(B) = x_{i_1}^{\sigma_1} \ldots x_{i_s}^{\sigma_s}$. Пусть B_1 и B_2 — два различных подкуба куба B^k. Пусть подкуб B_1 задается конъюнкцией $K(B_1)$, подкуб B_2 — конъюнкцией $K(B_2)$. Тогда через $\mathcal{N}(B_1, B_2)$ обозначим множество номеров переменных, которые входят и в конъюнкцию $K(B_1)$, и в конъюнкцию $K(B_2)$, причем в одну из конъюнкций с отрицанием, а в другую без.

Пусть $R(x_1, \ldots, x_k)$ — некоторая функция, определенная на k-мерном булевом кубе B^k. Произвольный набор $\mathcal{F} \in \{0, 1, 2\}^k$ будем называть фиксацией переменных $x_{i_1}, \ldots, x_{i_l}$, $l \leqslant k$, если для любого $i \in \{i_1, \ldots, i_l\} \subseteq \{1, 2, \ldots, k\}$ i-я компонента набора $\mathcal{F}$ не равна 2, а остальные компоненты этого набора равны 2. При этом i-ю компоненту $\mathcal{F}_i$ набора $\mathcal{F}$ будем называть фиксацией переменной x_i, если $\mathcal{F}_i \neq 2$. Через $\mathcal{F}(R)$ будем обозначать подфункцию, полученную из R подстановкой вместо всех переменных x_i, таких что $\mathcal{F}_i = 0$, нулей, и вместо всех переменных x_i, таких что $\mathcal{F}_i = 1$, единиц.

Заметим, что для решения поставленной задачи расшифровки функции f из $\Phi_{\mathcal{R}}^n$ достаточно построить алгоритм, который умеет определять функцию R граневого разбиения, $R \in \mathcal{R}$, и замену переменных $g \in G_n^{k(R)}$ такие, что $f = R(g(1), \ldots, g(k(R)))$.

Опишем алгоритм F, решающий поставленную задачу расшифровки функции $f : B^n \to \mathbb{N}$ из $\Phi_{\mathcal{R}}^n$, где $|\mathcal{R}| = m$ и $\max\limits_{i \in \{1,\ldots,m\}} k(R_i) = k$. Пусть $\mathcal{R}^{pos}$, $X^{R_1}, \ldots, X^{R_m}$ — вспомогательные множества, $ess^{R_1}, \ldots$, ess^{R_m} — вспомогательные наборы, которые будут использоваться в процессе работы алгоритма. Здесь

- $\mathcal{R}^{pos}$ — множество функций граневого разбиения из $\mathcal{R}$, которые на данном шаге работы алгоритма еще могут быть искомой функцией R граневого разбиения, то есть, такой функцией, что $f = R(g(1), \ldots, g(k(R)))$ для некоторой $g \in G_n^{k(R)}$. В начале работы алгоритма $\mathcal{R}^{pos} = \mathcal{R}$;
- ess^{R_i}, $i \in \{1, \ldots, m\}$, — набор длины $k(R_i)$, элементы которого могут принимать значения $y_1, \ldots, y_n, 0, 1$ и $*$. Если R_i — искомая

функция граневого разбиения, набор ess^{R_i} не содержит $*$, а $f = R_i(g(1), \ldots, g(k(R_i)))$, то $g(1) = ess_1^{R_i}, \ldots, g(k(R_i)) = ess_{k(R_i)}^{R_i}$. В начале работы алгоритма все элементы набора ess^{R_i} суть $*$;

- X^{R_i}, $i \in \{1, \ldots, m\}$, — множество номеров компонент набора ess^{R_i}, на данном шаге работы алгоритма не равных $*$;

Алгоритм F в процессе работы использует некоторую рекурсивную процедуру $S(A, l, r, lr, d)$, с описания которой мы и начнем. Здесь A — матрица размера $(u \times n)$ (u — некоторое натуральное число), l и r — некоторые переменные со значениями из $\mathbb{N}$. При каждом вызове процедуры S в процессе работы алгоритма F строки матрицы A будут являться наборами некоторой грани куба B^n, l и r будут равны значениям функии f на крайних наборах этой грани: на наборе с наибольшим числом нулей и на наборе с наибольшим числом единиц соответственно. Далее, lr — некоторая булева переменная, а d — некоторое множество целых чисел из $\{1, \ldots, u\}$. Процедура S рекурсивная и, как будет видно далее, в каждой неконечной итерации S вызывает себя дважды, назовем эти вызовы нулевым и первым. Переменная lr показывает, каким является текущий вызов процедуры S: нулевым ($lr = 0$) или первым ($lr = 1$). Множество d есть множество номеров строк матрицы A, на которых значение функции f известно к моменту текущего вызова процедуры S. Процедура S состоит в следующем.

Последовательно запрашиваем значения оператора $\mathcal{A}_f$ на строках матрицы A, номера которых не входят в множество d, до тех пор, пока не получим строку c, такую что $\mathcal{A}_f(c) \neq l$ и $\mathcal{A}_f(c) \neq r$. Возможны следующие варианты.

а) Такая строка c найдена, пусть она имеет номер q. Пусть d^l (d^r) — множество всех строк матрицы A, на которых запрошенное процедурой S значение оператора $\mathcal{A}_f$ оказалось равным l (соответственно r). Если $lr = 0$, полагаем $d^l = d \cup d^l$, если $lr = 1$, полагаем $d^r = d \cup d^r$. Далее полагаем $d^l = d^l \cup \{q\}$, $d^r = d^r \cup \{q\}$. Запускаем процедуру

$$S(A \cdot c, l, \mathcal{A}_f(c), 0, d^l).$$

Запускаем процедуру

$$S(A \vee c, \mathcal{A}_f(c), r, 1, d^r).$$

b) Такой строки не нашлось. Рассмотрим булев набор b длины u, такой что для любого $i \in \{1, \ldots, u\}$ $b_i = 0$, если $\mathcal{A}_f(A_i) = l$ и $b_i = 1$, если $\mathcal{A}_f(A_i) = r$, где A_i — i-я строка матрицы A. Для каждого $R \in \mathcal{R}^{pos}$ положим $j^R = \mathcal{N}(R^{-1}(l), R^{-1}(r))$. Исключаем из $\mathcal{R}^{pos}$ все функции R, такие что $j^R = \varnothing$. Находим в матрице A столбец, равный b (пусть это столбец с номером w), и для каждого $R \in \mathcal{R}^{pos}$ полагаем все элементы набора ess^R с номерами из множества j^R равными y_w.

Перейдем теперь к описанию собственно алгоритма F. Пусть A — произвольное правильное множество наборов из B^n минимальной мощности u, такое что матрица A не имеет ни нулевого, ни единичного столбца. Понятно, что $u =]\log_2(n+2)[$.

1) Запрашиваем значения $\mathcal{A}_f(0)$, $\mathcal{A}_f(1)$, где 0 — нулевой набор длины n, 1 — единичный набор длины n. Положим $l = \mathcal{A}_f(0)$, $r = \mathcal{A}_f(1)$. Если $l = r$, то искомая функция найдена — это тождественная функция, на всем B^n принимающая значение $l = r$. Исключаем из $\mathcal{R}^{pos}$ все функции, в области значений которых не входит l. Для каждой $R \in \mathcal{R}^{pos}$ рассматриваем конъюнкцию $K(R^{-1}(l))$. Если некоторая переменная с номером i входит в $K(R^{-1}(l))$ с отрицанием, полагаем $ess_i^R = 0$. Все остальные компоненты набора ess^R полагаем равными единице. Тогда, очевидно, для любого $R \in \mathcal{R}^{pos}$ выполнено $f = R(ess_1^R, \ldots, ess_{k(R)}^R)$, и алгоритм F заканчивает свою работу. Если же $l \neq r$, то запускаем процедуру $S(A, l, r, 0, \varnothing)$.
2) Пусть R — произвольный элемент множества $\mathcal{R}^{pos}$, такой что $k(R) \neq |X^R|$. Находим некоторую фиксацию $\mathcal{F}$ переменных с номерами из X^R, удовлетворяющую следующим свойствам:
   - для любого номера j из X^R, такого что $ess_j^R = 0$ ($ess_j^R = 1$) переменная с номером j фиксируется нулем (единицей), то есть $\mathcal{F}_j = 0$ ($\mathcal{F}_j = 1$);
   - для любых $i, j \in \{1, 2, \ldots, k\}$ если $ess_i^R = ess_j^R$, то $\mathcal{F}_i = \mathcal{F}_j$;

- для любого $j \in X^R$ выполнено $\mathcal{F}_j \neq 2$;
- функция $\mathcal{F}(R)$ существенно зависит хотя бы от одной переменной.

Если такой фиксации не нашлось, исключаем R из $\mathcal{R}^{pos}$ и переходим к 2. Через $h^{\mathcal{F}}$, $h \in \{0, 1, 2\}$, обозначим набор длины n, такой что его j-я компонента

$$h_j^{\mathcal{F}} = \begin{cases} \mathcal{F}_s, \text{ если } \exists s: \ ess_s^R = y_j; \\ h \text{ в противном случае.} \end{cases}$$

Если $\mathcal{A}_f(0^{\mathcal{F}}) \neq \mathcal{A}_f(1^{\mathcal{F}})$, запускаем процедуру

$$S(A \diamond 2^{\mathcal{F}}, \mathcal{A}_f(0^{\mathcal{F}}), \mathcal{A}_f(1^{\mathcal{F}}), 0, \varnothing)$$

и переходим к 2. В противном случае для каждой x_j, входящей в конъюнкцию $K(R^{-1}(\mathcal{A}_f(0^{\mathcal{F}})))$ с отрицанием (без отрицания), такой что $\mathcal{F}_j = 2$, полагаем $ess_j^R = 0$ ($ess_j^R = 1$ соответственно). Если существует такая функция $R \in \mathcal{R}^{pos}$, что $k(R) \neq |X^R|$, то переходим к 2.

3) Если $|\mathcal{R}^{pos}| = 1$, завершаем работу, так как в этом случае для единственной функции R, лежащей в $\mathcal{R}^{pos}$, выполнено $f = R(ess_1^R, \ldots, ess_{k(R)}^R)$. В противном случае пусть R_1 и R_2 — две произвольные функции из $\mathcal{R}^{pos}$, для которых существуют фиксация $\mathcal{F}'$ переменных $x_1, \ldots, x_{k(R_1)}$ и фиксация $\mathcal{F}''$ переменных $x_1, \ldots, x_{k(R_2)}$, удовлетворяющие следующим условиям:
   - если $ess_i^{R_1} = ess_j^{R_2}$ для некоторых i и j, то $\mathcal{F}'_i = \mathcal{F}''_j$;
   - для любого номера j из X^{R_1}, такого что $ess_j^{R_1} = 0$ ($ess_j^{R_1} = 1$) выполнено $\mathcal{F}'_j = 0$ ($\mathcal{F}'_j = 1$);
   - для любого номера j из X^{R_2}, такого что $ess_j^{R_2} = 0$ ($ess_j^{R_2} = 1$) выполнено $\mathcal{F}''_j = 0$ ($\mathcal{F}''_j = 1$);
   - $\mathcal{F}'(R_1)$ и $\mathcal{F}''(R_2)$ — константные функции, причем $\mathcal{F}'(R_1) \neq \mathcal{F}''(R_2)$.

Если таких функций R_1 и R_2 в $\mathcal{R}^{pos}$ нет, то завершаем работу, так как в этом случае для любой функции $R \in \mathcal{R}^{pos}$ выполнено $f = R(ess_1^R, \ldots, ess_{k(R)}^R)$. Если $\mathcal{A}_f(0^{\mathcal{F}'}) \neq \mathcal{F}'(R_1)$, исключаем

R_1 из $\mathcal{R}^{pos}$. Если $\mathcal{A}_f(0^{\mathcal{F}''}) \neq \mathcal{F}''(R_2)$, исключаем R_2 из $\mathcal{R}^{pos}$ Заметим, что множество всех переменных y_i, $i \in \{1, \ldots, n\}$, входящих в ess^{R_1}, и множество всех переменных y_j, $j \in \{1, \ldots, n\}$, входящих в ess^{R_2}, совпадают. Отсюда $\mathcal{A}_f(0^{\mathcal{F}'}) = \mathcal{A}_f(0^{\mathcal{F}''})$ и, значит, хотя бы одно из указанных выше неравенств выполняется. Переходим к 3.
На этом описание алгоритма завершено.

Перед тем, как перейти к доказательству корректности работы алгоритма, приведем пример его работы по расшифровке функции f, определяемой соотношениями (1). Здесь $f \in \Phi^5_{\mathcal{R}} = \Phi^5_{R_1} \cup \Phi^5_{R_2} \cup \Phi^5_{R_3}$, функции R_1, R_2, R_3 изображены на рис.1.

Согласно алгоритму выбираем правильное множество из $]\log_2 7[= 3$ наборов:

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Далее,

$$\mathcal{A}_f(00000) = 3 \Rightarrow l = 3;$$

$$\mathcal{A}_f(11111) = 4 \Rightarrow r = 4;$$

l и r суть разные числа, поэтому запускаем процедуру $S(A, 3, 4, 0, \varnothing)$. Запрашиваем значение $\mathcal{A}_f$ на первом наборе множества A:

$$\mathcal{A}_f(00011) = 4.$$

Так как $r = 4$, запрашиваем значение $\mathcal{A}_f$ на втором наборе:

$$\mathcal{A}_f(01100) = 2.$$

Поскольку $l \neq 2$, $r \neq 2$, полагаем $d^l = \varnothing \cup \{2\} = \{2\}$, $d^r = \{1\} \cup \{2\} = \{1, 2\}$, находим $A^0 = A \cdot (01100)$ и $A^1 = A \vee (01100)$:

$$A^0 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}; \quad A^1 = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

Запускаем процедуру $S(A^0, 3, 2, 0, \{2\})$. Поскольку $\mathcal{A}_f(A_1^0) = 3 = l$, $\mathcal{A}_f(A_2^0) = 2 = r$ (заметим, что здесь значение $\mathcal{A}_f(A_2^0)$ алгоритм не запрашивает, так как $2 \in \{2\}$, это значение алгоритму уже известно и равно $\mathcal{A}_f(A_2)$), $\mathcal{A}_f(A_3^0) = 2 = r$, получаем, что $b = (011)^T$. Далее имеем $j^{R_1} = \mathcal{N}(\overline{x}_3\overline{x}_1x_2, \overline{x}_3\overline{x}_1\overline{x}_2) = \{2\}$, $j^{R_2} = \mathcal{N}(\overline{x}_3x_1, \overline{x}_3\overline{x}_1) = \{1\}$, $j^{R_3} = \mathcal{N}(\overline{x}_2\overline{x}_1\overline{x}_3, \overline{x}_2\overline{x}_1x_3) = \{3\}$. Отсюда, учитывая, что b — третий столбец матрицы A^0, получаем

$$ess^{R_1} = (*, y_3, *, *),\ ess^{R_2} = (y_3, *, *),\ ess^{R_3} = (*, *, y_3).$$

Запускаем процедуру $S(A^1, 2, 4, 1, \{1, 2\})$. Поскольку $\mathcal{A}_f(A_1^1) = 4 = r$ (значение $A_f(A_1^1)$ не запрашивается, так как $1 \in \{1, 2\}$ и оно равно $A_f(A_1)$), $\mathcal{A}_f(A_2^1) = 2 = l$ (значение $A_f(A_2^1)$ не запрашивается, так как $2 \in \{1, 2\}$ и оно равно $A_f(A_2)$), $\mathcal{A}_f(A_3^1) = 4 = r$, получаем, что $b = (101)^T$. Далее имеем $j^{R_1} = \mathcal{N}(\overline{x}_3\overline{x}_1x_2, x_3\overline{x}_4) = \{3\}$, $j^{R_2} = \mathcal{N}(\overline{x}_3x_1, \overline{x}_2x_3) = \{3\}$, $j^{R_3} = \mathcal{N}(\overline{x}_2\overline{x}_1x_3, x_2) = \{2\}$. Отсюда, учитывая, что b — пятый столбец матрицы A^1, получаем

$$ess^{R_1} = (*, y_3, y_5, *),\ ess^{R_2} = (y_3, *, y_5),\ ess^{R_3} = (*, y_5, y_3).$$

Рассмотрим функцию $R_1(x_1, x_2, x_3, x_4)$. Согласно алгоритму, мы должны фиксировать переменные x_2 и x_3 так, чтобы функция R_1 не стала константной. Очевидно, такому требованию удовлетворяет фиксация $\mathcal{F}: \mathcal{F}_2 = 0,\ \mathcal{F}_3 = 1$. Тогда имеем $0^{\mathcal{F}} = (00001)$, $1^{\mathcal{F}} = (11011)$, откуда $\mathcal{A}_f(0^{\mathcal{F}}) = \mathcal{A}_f(1^{\mathcal{F}}) = 4$. Подкуб $R_1^{-1}(4)$ задается конъюнкцией $\overline{x}_4x_3$ и, учитывая, что $\mathcal{F}_3 \neq 2$, а $\mathcal{F}_4 = 2$, получаем, что переменная x_4 — константа 0. Отсюда $ess^{R_1} = (*, y_3, y_5, 0)$. Пусть теперь $\mathcal{F}: \mathcal{F}_2 = 1, \mathcal{F}_3 = 0$. Тогда $0^{\mathcal{F}} = (00100)$, $1^{\mathcal{F}} = (11110)$, откуда $\mathcal{A}_f(0^{\mathcal{F}}) = 2,\ \mathcal{A}_f(1^{\mathcal{F}}) = 5$, то есть, $\mathcal{A}_f(0^{\mathcal{F}}) \neq \mathcal{A}_f(1^{\mathcal{F}})$. Тогда $2^{\mathcal{F}} = (22120)$ и

$$A' = A \diamond 2^{\mathcal{F}} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Запускаем процедуру $S(A \diamond 2^{\mathcal{F}}, 2, 5, 0, \varnothing)$. Поскольку $\mathcal{A}_f(A_1') = 2 = l$, $\mathcal{A}_f(A_2') = 2 = l$, $\mathcal{A}_f(A_3') = 5 = r$, получаем, что $b = (001)^T$. Число

5 не входит в область значений функции R_2, потому исключаем ее из $\mathcal{R}^{pos}$. Для функций R_1 и R_3 имеем $j^{R_1} = \mathcal{N}(\overline{x}_3x_1, \overline{x}_3\overline{x}_1x_2) = \{1\}$, $j^{R_3} = \mathcal{N}(\overline{x}_2x_1x_3, \overline{x}_2\overline{x}_1x_3) = \{1\}$. Отсюда, учитывая, что b — первый столбец матрицы A', получаем

$$ess^{R_1} = (y_1, y_3, y_5, 0),\ ess^{R_3} = (y_1, y_5, y_3).$$

Осталось понять, какая функция — R_1 или R_2 — является искомой функцией граневого разбиения. Для этого согласно пункту 3 алгоритма выбираем две фиксации: $\mathcal{F}' : \mathcal{F}'_4 = 0,\ \mathcal{F}'_3 = 0,\ \mathcal{F}'_1 = 1,\ \mathcal{F}'_2 = 0$, $\mathcal{F}'' : \mathcal{F}''_3 = 0,\ \mathcal{F}''_1 = 1,\ \mathcal{F}''_2 = 0$. Тогда $\mathcal{F}''(R_2) = 2,\ 0^{\mathcal{F}'} = (10000)$ и $\mathcal{F}'(R_1) = 5 = \mathcal{A}_f(0^{\mathcal{F}'})$. Отсюда R_1 — искомая функция граневого разбиения, а функция $f = R_1(ess^{R_1}) = R_1(y_1, y_3, y_5, 0)$, как и было задумано.

Остановимся теперь на доказательстве корректности работы описанного алгоритма.

Лемма 2. *Для любой матрицы A и чисел l и r, подаваемых алгоритмом F на вход процедуры S существует такой номер i, $i \in \{1, \ldots, n\}$, что переменная y_i входит в $K(f^{-1}(l))$ с отрицанием и в $K(f^{-1}(r))$ — без, причем i-й столбец матрицы A не является ни единичным, ни нулевым.*

Доказательство. Заметим первым делом, что для l и r, подаваемых на вход S выполнено $l \neq r$. Если S вызывается из 1 или 2, то это свойство очевидно. Если же S сама себя вызывает, то нужное свойство очевидным образом доказывается по индукции.

Из определения f и неравенства $l \neq r$ следует, что подкубы $f^{-1}(l)$ и $f^{-1}(r)$ не пересекаются. Значит, существует переменная y_i, $i \in \{1, \ldots, n\}$, которая входит в одну из конъюнкций $K(f^{-1}(l))$, $K(f^{-1}(r))$ с отрицанием, а в другую — без.

Рассмотрим набор 0^A, j-я компонента которого равна единице, если j-й столбец матрицы A единичный, и нулю в противном случае, $j \in \{1, \ldots, n\}$. Аналогично j-я компонента набора 1^A равна нулю, если j-й столбец матрицы A нулевой, и единице в противном случае. Если процедура S вызвана из 1 или 2, то, очевидно, $\mathcal{A}_f(0^A) = l$, $\mathcal{A}_f(1^A) = r$. Действительно, при вызове S из 1 имеем $\mathcal{A}_f(0^A) = \mathcal{A}_f(0) = l$; $\mathcal{A}_f(1^A) = \mathcal{A}_f(1) = r$, при вызове S из 2

$\mathcal{A}_f(0^{A\diamond 2^{\mathcal{F}}}) = \mathcal{A}_f(0^{\mathcal{F}})$, $\mathcal{A}_f(1^{A\diamond 2^{\mathcal{F}}}) = \mathcal{A}_f(1^{\mathcal{F}})$. Покажем, что равенства $\mathcal{A}_f(0^A) = l$, $\mathcal{A}_f(1^A) = r$ верны и в случае, когда S сама себя вызывает. Для этого достаточно показать, что

$$\mathcal{A}_f(0^{A\cdot c}) = l,\ \mathcal{A}_f(1^{A\cdot c}) = \mathcal{A}_f(c),$$

если известно, что $\mathcal{A}_f(0^A) = l$. Требуемые равенства следуют из двух очевидных равенств $0^{A\cdot c} = 0^A,\ 1^{A\cdot c} = c$. Аналогично получаем, что из равенства $\mathcal{A}_f(1^A) = r$ следует, что

$$\mathcal{A}_f(0^{A\vee c}) = \mathcal{A}_f(c),\ \mathcal{A}_f(1^{A\vee c}) = r.$$

Положим, что y_i входит в $K(f^{-1}(l))$ с отрицанием, а в $K(f^{-1}(r))$ — без. Тогда, учитывая, что $\mathcal{A}_f(0^A) = l$, получаем равенство $0^A_i = 0$, а из равенства $\mathcal{A}_f(1^A) = r$ получаем, что $1^A_i = 1$. Таким образом, столбец матрицы A с номером i не может быть ни нулевым, ни единичным. То же утверждение мы бы получили, предположив, что y_i входит в $K(f^{-1}(l))$ без отрицания, а в $K(f^{-1}(r))$ — с отрицанием. Но на самом деле последний случай невозможен: поскольку все компоненты вектора 0^A, соответствующие неконстантным столбцам, нулевые, y_i входит в $K(f^{-1}(l))$ с отрицанием, аналогично y_i входит в $K(f^{-1}(r))$ без отрицания. Лемма доказана.

Лемма 3. *В пункте b) из описания процедуры S столбец матрицы A, равный b, существует и единственен.*

Доказательство. Существование. По определению набора b столбец с номером i, где y_i — переменная из леммы 2, равен b.

Единственность. Из доказательства леммы 2 следует, что столбец матрицы A с номером i не может быть ни единичным, ни нулевым. Осталось заметить, что все неединичные и ненулевые столбцы матрицы A попарно не совпадают в силу правильности исходного множества. Лемма доказана.

Лемма 4. *Пусть R — такая функция из $\mathcal{R}$, что $f \in \Phi^n_R$. Пусть в результате вызова процедуры S из 1 или 2 мощность множества X^R увеличивается на k'. Тогда количество N запросов значений оператора $\mathcal{A}_f$, сделанных процедурой S в этот вызов, не превосходит $k'(\rceil \log_2 n\lceil+2)$.*

Доказательство. Будем по указанному в условии Леммы вызову процедуры S строить бинарное дерево. Работа рекурсивной процедуры представляет собой некоторое число вызовов этой процедурой самой себя. Пусть изначально дерево состоит из одной вершины. Пометим ее 0. Сопоставим исходному вызову процедуры S эту вершину. Пусть далее некоторому вызову процедуры S с параметрами $S(A, l, r, lr, d)$ сопоставлена вершина v, причем в результате работы процедуры S в этот вызов она снова себя вызывает. Тогда добавим к дереву 2 вершины — левого и правого ребенка вершины v, левого ребенка сопоставим вызову S с параметрами $S(A \cdot c, l, \mathcal{A}_f(c), 0, d^l)$, правого — вызову S с параметрами $S(A \vee c, \mathcal{A}_f(c), r, 1, d^r)$. Левого ребенка пометим 0, правого — 1. Будем далее обозначать вложенный вызов процедуры, сопоставленный вершине v построенного дерева, через S^v. Через D обозначим множество всех вершин нашего дерева. Очевидно, что $N = \sum_{v \in D} N(S^v)$, где $N(S^v)$ — число запросов значений оператора $\mathcal{A}_f$ в течение вызова S^v процедуры S, здесь сумма берется по всем вершинам дерева. Пусть далее $N^l(S^v)$ — число запросов, на которые получен ответ l, $N^r(S^v)$ — число запросов, на которые получен ответ r.

Вершину v' с меткой 0 (меткой 1) будем называть прямым предком вершины v с меткой 0 (меткой 1 соответственно), если на пути между v и v' все вершины дерева имеют метку 0 (метку 1 соответственно). Для каждой концевой вершины v дерева через D^v обозначим множество всех ее прямых предков. Через D^{fin} обозначим множество концевых вершин нашего дерева, через D_l^{fin} — подмножество D^{fin}, каждый элемент которого имеет метку 0, D_r^{fin} — подмножество D^{fin}, каждый элемент которого имеет метку 1. Понятно, что для любых двух концевых вершин v_1 и v_2 $D^{v_1} \cap D^{v_2} = \varnothing$ и $\cup_{v \in D^{fin}} D^v = D$. Тогда

$$N = \sum_{v \in D^{fin}} \sum_{v' \in D^v} N(S^{v'}) \leqslant \sum_{v \in D^{fin}} \sum_{v' \in D^v} \Big(N^l(S^{v'}) + N^r(S^{v'}) \Big) + k' \leqslant$$

$$\leqslant \sum_{v \in D_l^{fin}} \left(\sum_{v' \in D^v} N^l(S^{v'}) + N^r(S^v) \right) + \sum_{v \in D_r^{fin}} \left(\sum_{v' \in D^v} N^r(S^{v'}) + N^l(S^v) \right) +$$

$$+ k' \leqslant (\rceil \log_2 n[+1) \left(\sum_{v \in D_l^{fin}} 1 + \sum_{v \in D_r^{fin}} 1 \right) + k' = k'(\rceil \log_2 n[+2).$$

Лемма доказана.

Следствие 1. *Пусть R — такая функция из $\mathcal{R}$, что $f \in \Phi_R^n$, k — число существенных переменных функции f. Тогда суммарное число значений оператора $\mathcal{A}_f$, запрошенное алгоритмом F через процедуру S для нахождения f, не превышает $k(\rceil \log_2 n[+2)$.*

Доказательство. Суммарно все запуски процедуры S могут увеличить мощность множества X^R не более чем на k, причем никакие два экземпляра процедуры S не добавляют в X^R одинаковых чисел. Отсюда, учитывая лемму 4, получаем утверждение следствия.

Лемма 5. *Пусть $k' = \max\limits_{i \in \{1,\ldots,m\}} k(R_i)$. Тогда суммарное число значений оператора $\mathcal{A}_f$, запрошенное алгоритмом F для нахождения функции f вне процедуры S, не превышает $2m(k'+1)$.*

Доказательство. Вне процедуры S алгоритм F запрашивает значения оператора $\mathcal{A}_f$ только в пунктах 2 и 3, когда сравнивает значения $\mathcal{A}_f(0^{\mathcal{F}})$ и $\mathcal{A}_f(1^{\mathcal{F}})$.

При каждом заходе в 2 алгоритм F делает не более двух запросов значений $\mathcal{A}_f$. Далее, при $\mathcal{A}_f(0^{\mathcal{F}}) \neq \mathcal{A}_f(1^{\mathcal{F}})$ вызывается процедура S, в результате чего мощность множества X^R возрастает по крайней мере на единицу. При $\mathcal{A}_f(0^{\mathcal{F}}) = \mathcal{A}_f(1^{\mathcal{F}})$ по крайней мере одна переменная входит в $K(R^{-1}(\mathcal{A}_f(0^{\mathcal{F}})))$ и, значит, мощность множества X^R снова возрастает по крайней мере на единицу. Отсюда, учитывая, что в X^R не может быть более k' элементов, получаем, что для каждого конкретного $R \in \mathcal{R}$ алгоритм запрашивает значения $\mathcal{A}_f$ не более $2k'$ раз. Вспоминая, что мощность $\mathcal{R}$ равна m, получаем результирующую оценку $2k'm$ для пункта 2.

Далее, поскольку при каждом заходе в 3 алгоритм F исключает из множества $\mathcal{R}^{pos}$ по крайней мере один элемент, а мощность этого множества не превосходит m, получаем для пункта 3 оценку $2m$. В сумме имеем $2k'm + 2m = 2m(k'+1)$. Лемма доказана.

Следствие 2. *Для любого множества функций граневого разбиения* $\mathcal{R} \in \boldsymbol{R}(m,k)$ *и для любой функции* $f \in \Phi^n_{\mathcal{R}}$ *имеет место неравенство*

$$\varphi(F,\mathcal{R},f) \leqslant k(\rceil \log_2 n[+2) + 2m(k+1).$$

Доказательство теоремы. Согласно лемме 1 и следствию 2 выполнено

$$(k-]\log_2 k[)]\log_2(n-k+1)[\leqslant \varphi(n,m,k) \leqslant k(\rceil \log_2 n[+2) + 2m(k+1),$$

из чего следует утверждение теоремы.

Список литературы

[1] Коробков В. К. О монотонных функциях алгебры логики // Проблемы кибернетики. Вып. 13. М.: Наука, 1965.

[2] Hansel G. О числе монотонных булевых функций n переменных // Кибернетич. сб. Новая серия. Вып. 5. М.: Мир, 1968.

[3] Шевченко В. Н. О расшифрковке пороговых функций многозначной логики // Комбинаторно-алгебраические методы в прикладной математике. Горький: Горьк. гос. ун-т, 1987. С. 155–163.

[4] Золотых Н. Ю. Расшифровка пороговых и близких к ним функций многозначной логики / Диссертация на соискание степени кандидата физико-математических наук. Нижегородский государственный университет, 1998.

[5] Кудрявцев В. Б., Гасанов Э. Э., Долотова О. А., Погосян Г. Р. Теория тестирования логических устройств. М.: ФИЗМАТЛИТ, 2006.

[6] Мошков М. Ю. Элементы математической теории тестов. Методические указания. Горький: ГГУ, 1986.

[7] Angluin D. Queries and Concept Learning // Machine Learning. Vol. 2. 1988. P. 319–342.

[8] Valiant L.G. A Theory of the Learnable // Comm. ACM. 27. 1984. P. 1134–1142.

[9] Осокин В. В. Асимптотика сложности разбиения булевого куба на подкубы // Материалы IX Международной конференции «Интеллектуальные системы и компьютерные науки». Т. 1. Ч. 2. 2006. С. 191–193.

[10] Осокин В. В. О сложности расшифровки разбиения булевого куба на подкубы // Дискретная математика, в печати.

[11] Осокин В. В. О расшифровке разбиения булевого куба на грани // Материалы IX Международного семинара «Дискретная математика и ее приложения», посвященного 75-летию со дня рождения академика О. Б. Лупанова. 2007. С. 343–346.