Расшифровка булевых функций фиксированного веса

Быстрыгова А.В.[1]

В работе исследуется сложность расшифровки класса булевых функций фиксированного веса при помощи запросов на значение, запросов на сравнение, запросов на ограниченную и расширенную эквивалентность. Причем, при расшифровке разрешено использовать только один из упомянутых типов запросов. Для всех типов запросов кроме запросов на сравнение получены точные оценки сложности расшифровки. Для запросов на сравнение приводится верхняя оценка, а также демонстрируется ее совпадение с нижней оценкой для функций веса 1, 2, 3.

Ключевые слова: булевы функции фиксированного веса, запросы на значение, запросы на сравнение, запросы на ограниченную эквивалентность, запросы на расширенную эквивалентность, точная расшифровка.

1. Введение

Должное внимание исследователей с середины прошлого века и по сей день приковано к задачам расшифровки функций фиксированными типами запросов. Под расшифровкой функции из заданного класса понимают игру между двумя игроками: учителем и учеником, — в которой учитель тайно выбирает одну функцию из класса, известного ученику, и затем отвечает на разные его вопросы в отношении выбранной функции. В свою очередь ученик, зная только класс функций, по ответам на свои вопросы должен восстановить функцию, загаданную учителем.

В теории расшифровки функций (computational learning theory) привычной практикой стало рассмотрение задачи расшифровки одного и

[1] *Быстрыгова Анастасия Викторовна* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: anastasiya.bistrigova@yandex.com.

Bistrigova Anastasiya Viktorovna — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

того же класса функций разными типами запросов или даже разными комбинациями нескольких типов запросов. Пожалуй, одной из первой работ, в которой встречаются запросы на значение, ограниченную эквивалентность и расширенную эквивалентность, является работа [1]. В своей работе Дана Ангуин приводит алгоритм расшифровки произвольного класса функций запросами на ограниченную эквивалентность, который по сути является переборным. В частности, с помощью него показано, что для расшифровки булевых функций веса 1 необходимо задать в худшем случае $2^n - 1$ запросов, причем похожий алгоритм можно адаптировать и под запросы на значение и получить такое же число запросов. Более того, автор приводит так называемый "алгоритм голосования большинством", позволяющий для произвольного класса функций F понять, какая функция выбрана учителем, за не более $\log_2 |F|$ запросов на расширенную эквивалентность. Автор приводит алгоритм расшифровки запросами на эквивалентность КНФ функции арности n, где каждая дизъюнкция состоит из не более k литерал. Также она показывает, что для расшифровки ДНФ функции арности n, в которой каждая конъюнкция состоит из одного литерала, в худшем случае требуется $2^n - 1$ запросов на эквивалентность. Помимо этого, она рассматривает задачу расшифровки монотонных ДНФ при помощи одновременного использования запросов на ограниченную эквивалентность и запросов на значение и показывает, что для расшифровки функции арности n с m элементарными конъюнкциями понадобится $m + 1$ запросов на ограниченную эквивалентность и mn запросов на значение. В другой своей работе [2] Англуин вводит обозначения для запросов на значение MQ, на ограниченную EQ и расширенную эквивалентность XEQ, которыми мы и будем далее пользоваться. Помимо этого, с помощью неравенств она демонстрирует соотношение между тем, насколько много запросов потребуется если использовать только один из упомянутых типов запросов или комбинации этих запросов независимо от класса функций.

Касаемо запросов на ограниченную и расширенную эквивалентность, многие работы направлены на обоснование существования полиномиальных алгоритмов расшифровки фиксированных классов функций с помощью этих типов запросов. Одной из таких работ стала работа [3], посвященная рассмотрению функций арности n, равных нулю в случае кратности p суммы значений переменных, и равных единице в противном случае. Авторам [3] удалось получить полиномиальный алгоритм расшифровки класса конъюнкций этих функций по модулю произвольного простого p. Количество запросов на ограниченную эквивалентность,

необходимых данному алгоритму, равно $n^{p-1}+1$. Авторы [4] продемонстрировали алгоритм, позволяющий расшифровать функцию из класса $\{\sum_{i=1}^{n} w_i x_i \geq N\}$ за $O(n^2 \log_2 n)$ запросов на ограниченную эквивалентность.

В работе [5] рассматривается расшифровка бесповторных функций в базисе всевозможных монотонных пороговых функций запросами на тождественность, которые по сути представляют собой комбинацию запросов на эквиваленность и запросов на значение. Запрос на тождественность — это подфункция. Если подфункция — это набор, тогда ответ на запрос есть значение функции в запрашиваемой точке, иначе, ответ равен 1, если подфункция тождественно равна константе, и 0, если это не так. Авторами показано, что для расшифровки бесповторной функции в указанном базисе требуется в худшем случае экспоненциального относительно числа переменных числа запросов тождественности, более того, для проивольного конечного подмножества этого базиса допускается решение полиномиальным числом запросов.

В данной работе помимо представленных Даной Англуин запросов на значение, запросов на ограниченную и расширенную эквивалентность рассматриваются также запросы на сравнение. Задача расшифровки запросами на сравнение стала исследоваться сравнительно недавно. Работа [6] стала первой, в которой изучается сложность расшифровки функций запросами на сравнение. В работе [7] автор рассматривал задачу расшифровки полиномиальных функций ранжирования. Похожий тип запросов применялся в задаче, рассматриваемой в работе [8]. Обе работы [9] и [10] посвящены получению оценок сложности расшифровки для всех замкнутых классов Поста. Различие в том, что в первой рассматривалась задача для запросов на значение, в то время как во второй — для запросов на сравнение. В работе [10] продемонстрировано, что для расшифровки замкнутых классов Поста запросы на сравнение не так уж и уступают запросам на значение, то есть для расшифрования функции из фиксированного замкнутого класса Поста необходимо примерно столько же запросов на сравнение, сколько необходимо запросов на значение. Более того, для двух классов $\{x\}, \{x, \overline{x}\}$ запросами на сравнение можно восстановить загаданную функцию быстрее, чем запросами на значение.

Данная работа посвящена исследованию сложности расшифровки функций фиксированного веса для каждого из четырех упомянутых выше запросов в отдельности: на значение, на сравнение, на ограниченную и расширенную эквивалентность. Для похожего класса — класса функций с ограниченным весом уже проведено исследование [11] функ-

ции Шеннона мощности плоских схем, реализующих такие функции. Но с точки зрении вопросов сложности расшифровки данный класс ранее никем не исследовался, если не брать в расчет результат Англуин для функций веса 1, приведенный выше.

2. Основные понятия и формулировка результатов

Под $P(n)$ арности n. Под $F(n,k)$, где $0 < k < 2^n$, будем понимать множество булевых функций арности n веса k, то есть функций, которые принимают значение 1 ровно на k наборах из 2^n двоичных наборов.

Через $f \equiv g$ при $f, g \in P(n)$ будем обозначать ситуацию, в которой для любого $a \in \{0,1\}^n$ верно равенство $f(a) = g(a)$.

Пусть загадана функция $f \in F(n,k)$. Тогда определим рассматриваемые типы запросов и ответы на них следующим образом.

Запросом на значение a к функции f называется набор a, а ответом на него является значение функции f на наборе из запроса $f(a)$.

Запросом на сравнение (a,b) будем называть упорядоченную пару наборов a, b, а под ответом на этот запрос понимать знак разности $f(a) - f(b)$.

Под *запросом на ограниченную эквивалентность g к функции f* принято считать функцию $g \in F(n,k)$, а под ответом на указанный запрос — слово YES, если $f \equiv g$, и любой набор b такой, что $f(b) \neq g(b)$, в противном случае.

Под *запросом на расширенную эквивалентность g* понимают функцию $g \in P(n)$, под ответом на этот запрос считают слово YES, если $f \equiv g$, и какой-то b такой, что $f(b) \neq g(b)$, в противном случае.

Будем говорить, что *последовательность запросов расшифровывает загаданную функцию $f \in F(n,k)$*, если последовательность конечна, состоит из запросов одного типа и функция f однозначно восстанавливается по ответам на запросы этой последовательности.

Алгоритмом расшифровки для запросов на значение $A^{MQ}_{n,k}$ будем называть процесс задания последовательности запросов на значение таким образом, что каждый элемент последовательности выбирается определенным образом в зависимости от ответов учителя на запросы — предыдущие члены последовательности, причем сформированная таким образом последовательность расшифровывают загаданную функцию $f \in F(n,k)$. Аналогично вводится определение алгоритма расшифровки для

запросов на ограниченную эквивалентность $A_{n,k}^{EQ}$ и расширенную эквивалентность $A_{n,k}^{XEQ}$, а также алгоритма расшифровки для запросов на сравнение $A_{n,k}^{CQ}$.

Через $\mathcal{A}_{n,k}^{MQ}$ будем обозначать множество всех алгоритмов $A_{n,k}^{MQ}$ расшифровки для запросов на значение. Похожим образом вводится определение множества всех алгоритмов расшифровки для запросов на ограниченную эквивалентность $\mathcal{A}_{n,k}^{EQ}$ и расширенную эквивалентность $\mathcal{A}_{n,k}^{XEQ}$, а также множества всех алгоритмов расшифровки для запросов на сравнение $\mathcal{A}_{n,k}^{CQ}$.

Через $q(A, f)$ обозначим минимальное количество первых запросов в последовательности запросов алгоритма A, которые расшифровывают функцию $f \in F(n, k)$. Тогда под *сложностью расшифровки запросами* будем понимать число запросов, которое придется задать наилучшему алгоритму расшифровки для расшифровки самой плохой функции. Иными словами, сложность расшифровки запросами типа $T \in \{MQ, CQ, EQ, XEQ\}$ задается следующим образом

$$\varphi_T(n, k) = min_{A \in \mathcal{A}_{n,k}^T} max_{f \in F(n,k)} q(A, f).$$

Будем считать, что ученику известны оба значения n и k.

Если a — вещественное число, тогда под $]a[$ будем понимать наименьшее целое, не меньшее a, под $[a]$ — наибольшее целое, не большее a, под $a \bmod b$ — остаток от деления a на b, под $|a|$ — модуль числа a.

Если A — множество, то под $|A|$ будем понимать мощность множества A.

Теорема 1. *Сложность расшифровки класса $F(n, k)$ запросами на значение равна $\varphi_{MQ}(n, k) = 2^n - 1$.*

Теорема 2. *Сложность расшифровки класса $F(n, k)$ запросами на ограниченную эквивалентность равна $\varphi_{EQ}(n, k) = 2^n - 1$.*

Теорема 3. *Сложность расшифровки класса $F(n, k)$ запросами на расширенную эквивалентность равна $\varphi_{XEQ}(n, k) = min(k, 2^n - k)$.*

Теорема 4. *Для запросов на сравнение справедлива следующая оценка $\varphi_{CQ}(n, 1) = 2^{n-1}$.*

Теорема 5. *Если $n \geq 2$, то для запросов на сравнение справедливо равенство $\varphi_{CQ}(n, 2) = [2^{n+1}/3]$.*

Теорема 6. *Если $n > 6$, то для запросов на сравнение справедливо следующее равенство* $\varphi_{CQ}(n,3) = 2^n -]3/2 \cdot [2^n/5][-[(2^n \bmod 5)/2]$.

Теорема 7. *Пусть $k \leq 2^{n-1}$ и для целых положительных $x_m, x_{m+1}, \ldots, x_{k-1}, x_k$, где $m =](k+1)/2[$, верно равенство*

$$2^n = m \cdot x_m + (m+1) \cdot x_{m+1} + \ldots + (k-1) \cdot x_{k-1} + k \cdot x_k.$$

Тогда справедлива следующая верхняя оценка

$$\varphi_{CQ}(n,k) \leq 2^n - (x_m + x_{m+1} + \ldots + x_{k-1} + x_k) + [max(x_m, x_{m+1}, \ldots, x_{k-1}, x_k)/2].$$

В частности, при подстановке определенных значений $x_m, x_{m+1}, \ldots, x_k$ в теорему 7, получаем следующую теорему.

Теорема 8. *Пусть $k \leq 2^{n-1}, m =](k+1)/2[, s = m + (m+1) + \ldots + (k-1) + k$, верно равенство $2^n = s \cdot q + r, r \in [0,s), q \geq m$, q, r — целые положительные числа. Тогда справедлива следующая верхняя оценка*

$$\varphi_{CQ}(n,k) \leq 2^n - (k-m+1)q - c + [0.5 \cdot max(q - r + (m+1)c, q + r - mc)] \leq 2^n - k/2 \cdot [2^n/s] + [0.5 \cdot ([2^n/s] +](k+1)/2[)],$$

где c вычисляется следующим образом

- $c = [2r/(2m+1)]$ *при* $2r \bmod (2m+1) \leq m$,
- $c = [2r/(2m-1)] + 1$ *при* $2r \bmod (2m+1) > m$.

3. Запросы на значение, ограниченную и расширенную эквивалентность

Приведем доказательство теоремы 1.

Доказательство. Верхняя оценка. Заметим, что в работе [1], было доказано неравенство $\varphi_{MQ}(n,1) \leq 2^n - 1$. Докажем данную верхнюю оценку для произвольного k. Запросим значение на любых $2^n - 1$ наборах. Если среди ответов на эти запросы встретилось ровно k единиц, значит значение функции на неопрошенном наборе равно 0, иначе 1. Соответственно, функция f восстановлена.

Нижняя оценка. Если ученик повторит запрос, просто ответим на него также, как отвечали прежне. Поэтому можно считать, что ученик не повторяет запросы. На первые $2^n - 2 - (k-1)$ запросов ученика будем отвечать 0, на следующие $k-1$ запросов ответим 1. Для ученика останутся неопрешенными 2 набора и не найдена одна единица загаданной функции. Поэтому он будет вынужден задать еще один запрос. □

Приведем доказательство теоремы 2.

Доказательство. Верхняя оценка. Заметим, что в работе [1], было доказано неравенство $\varphi_{EQ}(n,1) \leq 2^n - 1$. Докажем данную верхнюю оценку для произвольного k. Положим A — множество всех 2^n двоичных n-местных наборов, про которые мы пока не знаем, чему равно значение загаданной функции на них. A_0, A_1 — множество наборов, на которых значение загаданной функции равно 0 и 1 соответственно. Изначально, оба множества A_0, A_1 пусты. Каждый запрос формируем следующим образом. Выбираем любое подмножество B мощности $(k-|A_1|)$ из A и учителю передаем функцию g, которая равна единице на наборах из $B \cup A_1$ и равна нулю на остальных наборах (то есть наборах из $A_0 \cup (A \setminus B)$). Если учитель возвращает в ответ запрос x, то:

1) если $x \in B$, тогда удаляем x из A и добавляем в A_0,

2) если $x \notin B$, тогда удаляем x из A и добавляем в A_1.

Как только $|A| + |A_1| = k$, тогда делаем вывод, что загаданная функция равна единице на всех наборах в A и значит f восстановлена. Если $|A_1| = k$, тогда загаданная функция также восстановлена.

Достаточно задать $2^n - 1$ запросов. Так как за один запрос раскрывается значение ровно на одном наборе. Если $|A_1| < k$ после опроса $2^n - 1$ запросов, то значение функции на оставшемся наборе равно единице, иначе 0.

Нижняя оценка. Если в ответ на свой запрос ученик получил значение функции на каком-то наборе, а позже отправил запрос-функцию, которая отличается в соответствующем наборе от верного значения, тогда учитель в качестве ответа вновь отправляет ему этот набор.

Поэтому можно считать, что ученик не совершает такие “бесполезные“ запросы. На каждый из первых $2^n - 2 - (k-1)$ запросов будем возвращать набор, на котором функция ученика равна 1. Следовательно, за каждый такой запрос, мы раскроем информацию об одном нуле. На каждый из $k-1$ последующих запросов будем возвращать набор, на

котором функция ученика равна 0. Таким образом, за каждый такой запрос мы раскроем информацию об одной единице. Ученику неизвестно значение на ровно двух наборах и неизвестна одна единица. Поэтому он вынужден сделать еще один запрос. □

Далее следует доказательство теоремы 3.

Доказательство. Верхняя оценка. Пусть $k \leq 2^n - k$. Первым запросом отправим константу нуль. Так как $k > 0$, мы получим набор, на котором значение загаданной функции равно 1, тем самым мы раскроем информацию об одной единице. Далее отправим функцию, которая равна нулю всюду за исключением раскрытой единицы, и в ответ получим информацию о второй единице. Действуя дальше аналогично, за ровно k запросов мы восстановим загаданную функцию.

Если $k > 2^n - k$. Действуем аналогично с заменой 0 на 1 в предыдущей части доказательства.

Нижняя оценка. Пусть $k \leq 2^n - k$. Положим A — множество всех 2^n двоичных n-местных наборов, значение загаданной функции на которой пока неизвестно.

На каждый очередной запрос g ученика отвечаем следующим образом:

1) если в A лежит набор x, на котором g обращается в 1, тогда возвращаем ученику x и удаляем x из A,

2) иначе, если в A лежит набор x, на котором g обращается в 0, тогда возвращаем ученику x и удаляем x из A.

Если не сработал ни один из пунктов, значит множество A пустое, а значит вся функция f восстановлена.

Если учитель воспользовался первым пунктом своей стратегии, то он раскрыл ученику информацию об одном нуле, которых очень много.

Если учитель воспользовался вторым пунктом своей стратегии, то он раскрыл ученику информацию об одной единице, которых не так то и много.

Поэтому если ученик вынуждает учителя постоянно пользоваться пунктом 2, тогда он восстановит функцию за k запросов. □

4. Расшифровка функций веса 1 запросами на сравнение

Рассматривая запросы на сравнение, будем говорить, что *набор a был опрошен*, если он был одним из наборов какого-то заданного запроса на сравнение. Также будем говорить, *запрос (a, b) покрывает набор x*, если набор x совпадает с a или b. Будем говорить, что *множество запросов покрывает набор*, если хотя бы один запрос из этого множества покрывает этот набор.

Замечание 1. *Если известен ответ на запрос (a, b), а также значение функции на одном из наборов запроса ($f(a)$ или $f(b)$), тогда однозначно восстанавливается значение функции на втором наборе запроса ($f(b)$ или $f(a)$ соответственно).*

Лемма 1. *Справедлива следующая верхняя оценка*

$$\varphi_{CQ}(n, k) \leq k[2^n/(k+1)] + max(0, (2^n \bmod (k+1)) - 1).$$

Доказательство. Пусть $2^n = (k+1)q + r, 0 \leq r < (k+1), 0 < q$, q, r — целые числа.

Упорядочим произвольным образом все 2^n двоичных наборов и обозначим их через $x_0, x_1, \ldots, x_{2^n-2}, x_{2^n-1}$. Зададим q групп по k запросов. i-я $(0 \leq i < q)$ группа состоит из запросов вида $(x_{i\cdot(k+1)+j}, x_{i\cdot(k+1)+j+1})$, где j меняется от 0 до $k-1$ включительно.

Рассмотрим ответы для i-й группы запросов $(0 \leq i < q)$. Возможны два случая.

1) Ответы все запросы равны 0, тогда значение загаданной функции на всех наборах из запросов одно и то же. В силу того, что таких наборов $k+1$, а единиц функции ровно k, значит единицами функциями эти наборы не могут быть и $f(x_{i\cdot(k+1)+j}) = 0$ для всех $j \in \{0, 1, \ldots, k\}$.

2) Существует j_0 — наименьшее целое из интервала $[0, k-1]$ такое, что ответ на запрос отличен от нуля, иными словами, $f(x_{i\cdot(k+1)+j_0}) \neq f(x_{i\cdot(k+1)+j_0+1})$.

   a) Если ответ на запрос $(x_{i\cdot(k+1)+j_0}, x_{i\cdot(k+1)+j_0+1})$ равен 1, то в силу того, что ответы на все запросы $(x_{i\cdot(k+1)+j}, x_{i\cdot(k+1)+j+1}), j < j_0$, равны 0, а $f(x_{i\cdot(k+1)+j_0}) = 1$, следует $f(x_{i\cdot(k+1)+j}) = 1$ для

всех $j \in \{0, 1, \ldots, j_0\}$. Более того, из ответов на оставшиеся запросы этой группы можно последовательно восстановить значение на всех наборах этой группы, то есть сначала восстановить значение на наборе $x_{i\cdot(k+1)+j_0+1}$, затем на наборе $x_{i\cdot(k+1)+j_0+2}$ и так далее. Это возможно в силу замечания 1.

б) Если ответ на запрос $(x_{i\cdot(k+1)+j_0}, x_{i\cdot(k+1)+j_0+1})$ равен -1, то в силу того, что ответы на все запросы $(x_{i\cdot(k+1)+j}, x_{i\cdot(k+1)+j+1}), j < j_0$, равны 0, а $f(x_{i\cdot(k+1)+j_0+1}) = 1$, следует $f(x_{i\cdot(k+1)+j}) = 0$ для всех $j \in \{0, 1, \ldots, j_0\}$. Аналогично предыдущему пункту восстановим значение на всех остальных наборах этой группы.

Итого, будет задано $kq = k \cdot [2^n/(k+1)]$ запросов и восстановленно значение на всех $q \cdot (k+1)$ наборах, останется неизвестным значение на r наборах. Обозначим через x количество единиц, которое будет найдено за эти kq запросов. Тогда если $x = k$, то все единицы уже найдены. Если $x + r = k$, то оставшиеся r наборов также являются единицами и значит вновь функция полностью расшифрована. Иначе, необходимо найти $k - x$ единиц, где $0 < k - x < r$. Зададим $r - 1$ запросов, где в каждом запросе первая компонента — это один из непокрытых r наборов, а вторая — любой из покрытых ранее. Тогда по ответам на эти запросы мы однозначно восстановим значение функции на всех наборах, кроме оставшегося непокрытого одного. Если после этих запросов найдены все единицы искомой функции, значит непокрытый набор является нулем функции, иначе, этот набор и есть оставшаяся ненайденная единица.

В общей сложности для расшифровки функции будет потрачено следующее количество запросов.

1) Если $r = 0$, то $k \cdot 2^n/(k+1)$ запросов.

2) Если $r \geq 1$, то $k \cdot [2^n/(k+1)] + (r-1)$ запросов.

□

Приведем доказательство теоремы 4.

Доказательство. Покажем, что $\varphi_{CQ}(n, 1) \geq 2^{n-1}$. Пусть ученик задаст $2^{n-1} - 1$ запросов, на каждый ответим 0. В итоге, суммарно будет опрошено не более $(2^{n-1} - 1) \cdot 2 = 2^n - 2$ наборов. Про каждый из них ученик поймет, что значение функции на нем равно 0. Остается как минимум два набора, на которых неизвестно значение функции, следовательно нужно задать хотя бы еще один запрос.

Учитывая верхнюю оценку, полученную в лемме 1, получаем оценку $\varphi_{CQ}(n,1) = 2^{n-1}$. □

5. Расшифровка функций веса 2 запросами на сравнение

Прежде всего, рассмотрим следующую задачу. Имеется n одноэлементных множеств. За одну операцию разрешается объединить в одно множество любые два имеющихся на данный момент множества. Цель — за наименьшее число операций объединения получить множества мощности не меньшей заданного числа.

Заметим, что эту задачу можно переформулировать следующим образом. Изначально имеется n-вершинный граф без ребер. За одну операцию можно провести неориентированное ребро между двумя вершинами. Необходимо, использовав минимальное число ребер, объединить исходные n вершин в компоненты связности, в которых вершин не меньше заданного числа.

Лемма 2. *Пусть n кратно q, тогда для того, чтобы после операций объединения остались только множества мощности строго равной q, достаточно в точности $n \cdot (q-1)/q$ операций, более того, эта оценка не понижаема.*

Доказательство. Поскольку все n элементов распадутся на множества мощности равной q, тогда получится в точности n/q множеств. Чтобы объединить q элементов в одно множество необходимо в точности $q-1$ операций объединения, соответствующее количеству ребер в дереве из q вершин. □

Лемма 3. *Пусть n кратно q, тогда минимальное количество операций объединения, которое необходимо выполнить для того, чтобы после операций объединения остались только множества мощности не меньше q, равно $n \cdot (q-1)/q$.*

Доказательство. Согласно лемме 2, если объединять множества так, чтобы получились только множества мощности ровно q, необходимо не менее $n \cdot (q-1)/q$ операций.

Рассмотрим общий случай итоговой системы множеств после применения операций объединения. Пусть n_1 кратно q, n_2 кратно q_2, …, n_r кратно q_r, при этом $q < q_2 < \ldots < q_r, n_1 + n_2 + \ldots + n_r = n$. Тогда для

того, чтобы n_1 изначальных одноэлементных множеств объединились в q-элементные, n_2 — в q_2-элементные, ..., n_r — в q_r-элементные множества, потребуется, согласно лемме 2, не менее $n_1(q-1)/q+n_2(q_2-1)/q_2+\ldots+n_r(q_r-1)/q_r$ операций объединения.

Но учитывая, неравенство $\frac{t_1-1}{t_1} < \frac{t_2-1}{t_2}$ при $0 < t_1 < t_2$, получаем цепочку неравенств

$$n\frac{q-1}{q} - (n_1\frac{q-1}{q} + n_2\frac{q_2-1}{q_2} + \ldots + n_r\frac{q_r-1}{q_r}) <$$
$$< n\frac{q-1}{q} - \frac{q-1}{q}(n_1+n_2+\ldots+n_r) = \frac{q-1}{q}\cdot(n-n) = 0.$$

Следовательно, $n\frac{q-1}{q} < (n_1\frac{q-1}{q} + n_2\frac{q_2-1}{q_2} + \ldots + n_r\frac{q_r-1}{q_r})$.

Иными словами, для объединения изначальных n одноэлементных множеств во множества мощности не меньше q оптимальнее всего объединить их во множества мощности равной q. □

Лемма 4. *В графе с V вершинами, E ребрами и K компонентами связности справедливо неравенство $E+K \geq V$.*

Доказательство. Доказательство по индукции по количеству ребер в графе.

База индукции. $E = 0$, тогда $K = V$. Очевидно неравенство $K \geq V$. Переход индукции. Пусть $E > 0$ и справедливо неравенство $E + K \geq V$. Добавим одно ребро, тогда E увеличится на 1, а K либо уменьшится на 1, если новое ребро связало вершины из разных компонент связности, либо не изменится, если новое ребро связало две вершины из одной компоненты связности. Следовательно, неравенство $E + K \geq V$ сохранится. □

Лемма 5. *Пусть $n = xq + r$, где q, x, r — целые положительные числа, $1 \leq r < q$, тогда для того, чтобы после операций объединения остались только множества мощности не меньше q, необходимо сделать не менее $x \cdot (q-1) + r$ операций объединения.*

Доказательство. Докажем от противного. Пусть существуют n, q, x, r — целые положительные, такие что $n = xq + r, 0 < r < q$, и существует порядок объединения исходных n множеств во множества мощности не менее q, в котором используется $x\cdot(q-1)+(r-1)$ операций объединения.

Согласно лемме 4, исходные n вершин после добавления $x \cdot (q-1) + (r-1)$ ребер распадутся на не менее $n - (x \cdot (q-1) + r - 1) = x + 1$

компонент связности. При этом, в текущей лемме выше утверждается, что все получившиеся компоненты связности имеют мощность не менее q. Соответственно, суммарно во всех получившихся компонентах связности не менее $q(x+1) = qx + q$ вершин. Получили противоречие с исходным количеством вершин $n = qx + r < qx + q$. □

Приведем доказательство теоремы 5.

Доказательство. Докажем нижнюю оценку $\varphi_C(n, 2) \geq 2[2^n/3] + (2^n \bmod 3) - 1$. Возможны два случая: $2^n = 3x + 1$ и $2^n = 3x + 2$, где x — целое.

Рассмотрим первый случай: $2^n = 3x + 1$. Необходимо доказать, что ученик задаст хотя бы $2x$ запросов. Пусть он задаст $2x - 1$ запросов, на каждый запрос ответим 0, тогда покажем, что либо ему недостаточно информации, полученной по этим запросам, чтобы восстановить загаданную функцию, либо по заданным запросам можно восстановить порядок объединения множеств во множества мощности не менее 3 за меньшее число запросов, чем утверждается в нижней оценке в лемме 3 или лемме 5. Итак, ученик задал $2x - 1$ запросов. Рассмотрим все возможные случаи.

1) $2x - 1$ запросов покрывают не более $3x$ наборов. Обозначим через a_0 один из непокрытых наборов. Согласно лемме 3, $3x$ наборов после не менее $2x$ запросов могут объединиться в x множеств мощности 3. Но раз задано на один запрос меньше, то в лучшем случае сформировано уже $x - 1$ множеств мощности 3, а для образования еще одного такого не хватает одного запроса. Следовательно, среди всех наборов, за исключением набора a_0, имеется еще

   а) либо три непокрытых набора a_1, a_2, a_3,

   б) либо один непокрытый a_1 и одно множество, состоящее из наборов a_2, a_3,

   в) либо два двухэлементных множества.

   В первом случае 2 единицы функции могут находиться в любых двух наборах из a_0, a_1, a_2, a_3. Во втором случае 2 единицы либо a_2, a_3, либо a_0, a_1. В третьем случае, обе единицы могут находиться в любом из этих двух двухэлементных множеств.

   В любом случае, ученик вынужден задать еще хотя бы один запрос, чтобы избавиться от возникшей неоднозначности.

2) $2x-1$ запросов покрывают все $2^n = 3x+1$ наборов. В силу того, что отсутствуют непокрытые наборы, ответы на все запросы равны 0, значит все наборы разбились на множества мощности хотя бы 2. Если имеются как минимум два множества мощности ровно 2, то учитель может спрятать обе единицы в одно из множеств, поэтому ученик вынужден задать еще хотя бы один запрос. Следовательно, либо множеств мощности 2 ровно одно или нуль.

   Случай отсутствия множеств мощности 2 невозможен, так как в этом случае все $3x+1$ одноэлементных множеств за $2x-1$ операций объединились во множества мощности не менее 3, что противоречит лемме 5.

   Следовательно, после $2x-1$ операций объединения исходные $3x+1$ одноэлементных множества объединились во множества, среди которых ровно одно двухэлементное, а остальные мощности не менее трех. Но и этот случай невозможен, так как противоречит лемме 5, потому что $(3x+1)-2 = 3(x-1)+2$ одноэлементных множеств объединились во множества мощности не менее 3 за $(2x-1)-1$ операций вместо $2x$, как утверждается в лемме 5.

Следовательно, в случае $2^n = 3x+1$ ученик вынужден задать как минимум $2x$ запросов.

Рассмотрим случай $2^n = 3x+2$. Необходимо доказать, что ученик вынужден задать хотя бы $2x+1$ запросов.

Пусть ученик задал $2x$ запросов. На первые $2x-1$ запросов отвечаем 0.

Если $2x$ запросов покрывают ровно $3x$ наборов и к запросу с номером $2x$ в точности $3x-3$ наборов объединились во множества мощности 3, а запрос с номером $2x$, если в ответ на него придет 0, объединит еще три элемента в одно трехэлементное множество, то есть одна компонента этого запроса — набор одноэлементного множества, вторая — набор двухэлементного множества, тогда ответим

- -1, если первая компонента — набор одноэлементного множества,
- 1, если вторая компонента — набор одноэлементного множества.

Иначе, на запрос с номером $2x$ ответим 0.
Рассмотрим все возможные случаи.

1) $2x$ запросов покрыли не более $3x-1$ наборов. Тогда хотя бы 3 набора не покрыты. Ответы на все запросы были равны 0. Учитель

может спрятать обе единицы в двух из трех непокрытых наборах. Поэтому ученик вынужден задать хотя бы еще один запрос.

2) $2x$ запросов покрыли ровно $3x$ наборов. Тогда возможна одна из следующих двух ситуаций.

   а) Ответ на запрос с номером $2x$ был отличен от 0. Следовательно, учитель раскрыл информацию о ровно одной единице и двух нулях функции. Осталось найти еще одну единицу. В силу того, что ответы на первые $2x - 1$ запросов были равны 0, то все остальные наборы из рассматриваемых $3x$ объединились во множества мощности не меньшей 2. В этих множествах вторая единиица не может находиться, значит она находится среди непокрытых двух наборов. Следовательно, ученик вынужден задать еще один запрос для однозначного восстановления функции.

   б) Ответ на запрос с номером $2x$ был равен 0. Следовательно, рассматриваемые $3x$ наборов не объединились в x множеств мощности 3. Возможна одна из следующих ситуаций.

      – Среди множеств, в которые объединились $3x$ наборов, есть множество мощности 2. Тогда учитель может спрятать обе единицы либо в два непокрытых набора, либо в этом двухэлементном множестве. Ученик вынужден задать еще один запрос для восстановления функции.

      – Среди множеств, в которые объединились $3x$ наборов, нет множеств мощности 2. Но не все множества мощности ровно 3. Следовательно, $3x$ наборов объединились в p множеств мощности не менее 3, причем существует хотя бы одно множество мощности строго больше 3. На образование p множеств мощности ровно 3 необходимо не менее $2p$ запросов, и суммарно останется $3x - 3p > 0$ наборов раскидать по этим множествам, на добавление каждого такого набора в какое-то из p множеств тратится не менее 1 запроса. Исходя из этого, получаем цепочку неравенств $2x \geq 2p + (3x - 3p)$, $p \geq x$, $3p \geq 3x$, что противоречит неравенству $3x - 3p > 0$.

3) $2x$ запросов покрыли хотя бы $3x + 1$ наборов. Тогда возможна одна из следующих трех ситуаций.

а) Не менее $3x+1$ наборов объединились во множества мощности не меньшей 3. Этот случай невозможен, так как противоречит лемме 5.

б) Среди множеств, в которые объединились не менее $3x+1$ наборов, имеются хотя бы два множества мощности 2. Тогда обе единицы можно спрятать в любое из них и ученик вынужден использовать еще один дополнительный запрос для понимания, какое из двухэлементных множеств содержит обе единицы.

в) Среди множеств, в которые объединились не менее $3x+1$ наборов, имеется ровно одно множество мощности 2. Для рассмотрения этой ситуации отдельно рассмотрим случай, когда $2x$ запросов покрыли $3x+1$ наборов и когда $2x$ запросов покрыли $3x+2$ набора.

- $2x$ запросов покрыли $3x+1$ наборов. Соответственно, $3x-1 = 3(x-1)+2$ наборов за $2x-1 = 2(x-1)+1$ запросов объединились во множества мощности не менее 3, что противоречит лемме 5. Значит, этот случай невозможен.
- $2x$ запросов покрыли $3x+2$ набора. Получается, что $3x$ наборов за $2x-1$ запросов объединились во множества мощности не менее 3, что противоречит лемме 3. Следовательно, и этот случай невозможен.

Исходя из рассмотренных случаев становится ясно, что в случае $2^n = 3x+2$ ученик вынужден задать как минимум $2x+1$ запросов.

Учитывая нижнюю оценку, полученную в доказательстве данной теоремы, и верхнюю оценку, полученную в лемме 1, имеем следующее равенство $\varphi_{CQ}(n,2) = 2[2^n/3] + (2^n \bmod 3) - 1$.

Покажем, что последняя величина равна $[2^{n+1}/3]$. Для этого рассмотрим два случая.

1) Если $2^n = 3x+1$, тогда $\varphi_{CQ}(n,2) = 2x$, но с другой стороны $[2^{n+1}/3] = [2(3x+1)/3] = [2x+2/3] = 2x$.

2) Если $2^n = 3x+2$, тогда $\varphi_{CQ}(n,2) = 2x+1$, но с другой стороны $[2^{n+1}/3] = [2(3x+2)/3] = [2x+4/3] = 2x+1$.

□

6. Верхние оценки сложности расшифровки запросами на сравнение

Далее рассмотрим доказательство теоремы 7, представляющую верхнюю оценку сложности расшифровки функций любой арности n и произвольного веса $k \leq 2^{n-1}$.

Доказательство. Напомним, что через m мы обозначали $](k+1)/2[$. Произвольным образом разобьем 2^n наборов на x_m множеств размера m, x_{m+1} множеств размера $m+1$, ..., x_{k-1} множеств размера $k-1$ и x_k множеств размера k. Заметим, что $k-m < m$. Обозначим все множества следующим образом $A_{i,j}$, где i указывает размер множества, то есть число от m до k, а j — номер множества размера i, то есть число от 1 до x_i. Элементы множества $A_{i,j}$ будем обозначать $\{a^h_{i,j} | h = 1, i\}$.

Для каждого множества $A_{i,j}, i = \overline{m,k}, j = \overline{1,x_i}$ зададим следующие запросы: $(a^1_{i,j}, a^2_{i,j}), (a^2_{i,j}, a^3_{i,j}), \ldots, (a^{i-2}_{i,j}, a^{i-1}_{i,j}), (a^{i-1}_{i,j}, a^i_{i,j})$. Иными словами, мы как будто сцепляем изолированные i вершин ребрами в одну цепочку. Рассмотрим ответы на эти $i-1$ запросов. Возможны два случая.

1) Ответы все запросы равны 0, тогда значение загаданной функции на всех наборах множества $A_{i,j}$ одно и то же.

2) Существует q — наименьшее целое из интервала $[1, i-1]$ такое, что ответ на q-й запрос отличен от нуля, иными словами, $f(a^q_{i,j}) \neq f(a^{q+1}_{i,j})$.

   а) Если ответ на запрос $(a^q_{i,j}, a^{q+1}_{i,j})$ равен 1, то в силу того, что ответы на все запросы $(a^p_{i,j}, a^{p+1}_{i,j}), p < q$ равны 0, а $f(a^q_{i,j}) = 1$, следует $f(a^p_{i,j}) = 1$ для всех $p \in \{1, 2, \ldots, q-1, q\}$. Более того, из ответов на оставшиеся запросы этой группы можно последовательно восстановить значение на всех наборах этой группы, то есть сначала восстановить значение на наборе $a^{q+1}_{i,j}$, затем на наборе $a^{q+2}_{i,j}$ и так далее. Это осуществимо благодаря замечанию 1.

   б) Если ответ на запрос $(a^q_{i,j}, a^{q+1}_{i,j})$ равен -1, то в силу того, что ответы на все запросы $(a^p_{i,j}, a^{p+1}_{i,j}), p < q$ равны 0, а $f(a^{q+1}_{i,j}) = 1$, следует $f(a^p_{i,j}) = 0$ для всех $p \in \{1, 2, \ldots, q-1, q\}$. Аналогично предыдущему пункту восстановим значение на всех остальных наборах этой группы.

Обратим внимание на то, что в результате этих запросов про каждое множество $A_{i,j}$ мы знаем либо значение функции на всех наборах этого множества, либо то, что на всех наборах множества функция принимает одинаковое значение.

Подытоживая, получаем, что было задано $(m-1)\cdot x_m + m\cdot x_{m+1} + \ldots + (k-2)\cdot x_{k-1} + (k-1)\cdot x_k$ запросов, все 2^n наборов покрыты. Причем про какие-то наборы уже известно, чему равно значение функции в них, а оставшиеся наборы распались на классы эквивалентности мощности не меньшей m. Обозначим через t количество единиц функции, найденных в результате этих запросов. Если $t = k$, то искомая функция расшифрована. Иначе, осталось найти $k - t > 0$ единиц и точно известно, что все они лежат ровно в одном из образовавшихся множеств — классов эквивалентности. Поскольку в случае, если бы все ненайденные единицы находились хотя бы в двух множествах A_{i_1,j_1}, A_{i_2,j_2}, тогда получилось бы, что $k - t \geq i_1 + i_2 > k$.

Следовательно, если после получения ответов на упомянутые выше запросы, не были найдены все единицы, то остается найти $t \in [m,k]$ единиц, которые находятся ровно в одном из множеств $A_{t,1}, A_{t,2}, \ldots, A_{t,x_t-1}, A_{t,x_t}$. Эта задача эквивалентна задаче поиска одной единицы среди наборов $a^1_{t,1}, a^1_{t,2}, \ldots, a^1_{t,x_t-1}, a^1_{t,x_t}$, то есть среди первых элементов этих множеств. Причем, про какие-то из этих множеств мы уже узнали значение функции на них. Не нарушая общности, будем считать, что про последние $(x_t - p) \in [0, x_t - 1]$ множеств $A_{t,p+1}, A_{t,p+2}, \ldots, A_{t,x_t}$ мы знаем значение функции на каждом его элементе. Соответственно, необходимо найти единицу среди наборов $a^1_{t,1}, a^1_{t,2}, \ldots, a^1_{t,p}$.

Для этого понадобится $[p/2]$ запросов вида $(a^1_{t,2k+1}, a^1_{t,2k+2})$, где $k = \overline{0, [p/2]-1}$. Если $[p/2]$ — нечетно, то одна из единиц будет точно набор $a_{t,p}$, если на все последние запросы будет получен ответ 0.

Итого, в худшем случае будет задано $(m-1)\cdot x_m + m\cdot x_{m+1} + \ldots + (k-2)\cdot x_{k-1} + (k-1)\cdot x_k + [x_t/2]$ запросов. Из равенства $2^n = m\cdot x_m + (m+1)\cdot x_{m+1} + \ldots + (k-1)\cdot x_{k-1} + k\cdot x_k$ получаем равенство

$$(m-1)\cdot x_m + m\cdot x_{m+1} + \ldots + (k-2)\cdot x_{k-1} + (k-1)\cdot x_k = \\ = 2^n - (x_m + x_{m+1} + \ldots + x_k).$$

Это и приводит нас к оценке из теоремы. □

Перейдем к доказательству теоремы 8.

Доказательство. Положим $x_m = q - r + (m+1) \cdot c$, $x_{m+1} = q + r - m \cdot c$, а $x_{m+2} = x_{m+3} = \ldots = x_k = q$. Обозначим через $p = 2r \bmod (2m+1)$. Определим c следующим образом.

- при $p \leq m$ положим равным $c = [2r/(2m+1)]$,
- при $p > m$ положим равным $c = [2r/(2m+1)] + 1$.

Заметим, что в случае $p \leq m$ выполнено $x_{m+1} - x_m = 2r - c(2m+1) = p \leq m$. В случае $p > m$ верна цепочка равенств $x_m - x_{m+1} = c(2m+1) - 2r = 2m+1-p \leq m$. Таким образом, мы присвоили x_i значения, отличающиеся по модулю не более чем на m.

Подставляя полученные значения x_i в верхнюю оценку теоремы 7, получаем первое неравенство теоремы.

Заметим, что $k-m+1 \geq k/2$, $q = [2^n/s]$. Учитывая неравенство, $|x_m - x_{m+1}| \leq m$, получаем $max(x_m, x_{m+1}) \leq q+m = [2^n/s]+](k+1)/2[$. Все это позволяет нам получить второе неравенство из утверждения теоремы. □

7. Расшифровка функций веса 3 запросами на сравнение

Лемма 6. *Пусть $n \geq 3$ и для целых положительных x_2, x_3 верно равенство $2^n = 2x_2 + 3x_3$, причем $x_2 \geq 4$. Тогда справедливо неравенство*

$$2[(2^n - 3x_3)/3] + ((2^n - 3x_3) \bmod 3) - 1 \leq x_2 + [(x_2 - 1)/2].$$

Доказательство. Пусть $2^n - 3x_3 = 3q + r$, где q, r — целые неотрицательные числа, $r \in [1, 2]$. Соответственно, имеем цепочку равенств $2x_2 = 2^n - 3x_3 = 3q + r$. Перепишем искомое неравенство при помощи введенных обозначений $2[2x_2/3] + (2x_2 \bmod 3) - 1 \leq x_2 + [(x_2 - 1)/2]$ или $2q + r - 1 \leq x_2 + [(x_2 - 1)/2]$. Рассмотрим два случая в зависимости от делимости x_2 на 2.

- Пусть $x_2 = 2s, s \geq 0$. Левая часть искомого неравенства имеет вид $(2q + r) - 1 = (3q + r) - q - 1 = 4s - (4s - r)/3 - 1 = 8s/3 + r/3 - 1$. Правая часть искомого неравенства имеет вид $2s + (s - 1) = 3s - 1$. Рассмотрим разность $8s/3 + r/3 - 1 - 3s + 1 = -s/3 + r/3$. Приходим к выводу, что искомое неравенство справедливо при $s \geq 2, x_2 \geq 4, (2^n - 3x_3) \geq 8, n \geq 3$.

- Пусть $x_2 = 2s + 1, s \geq 0$. Левая часть искомого неравенства имеет вид $(2q + r) - 1 = (3q + r) - q - 1 = 4s + 2 - (4s + 2 - r)/3 - 1 = 8s/3 + (1 + r)/3$. Правая часть искомого неравенства имеет вид $2s + 1 + s = 3s + 1$. Рассмотрим разность $8s/3 + (1 + r)/3 - 3s - 1 = -s/3 - 2/3 + r/3$. Приходим к выводу, что искомое неравенство справедливо при $s \geq 0$, что у нас изначально и выполняется.

□

Лемма 7. *Пусть $n \geq 3$ и для целых положительных x_2, x_3 верно равенство $2^n = 2x_2 + 3x_3$, причем $x_2 < 4 \leq x_3$. Тогда имеет неравенство*

$$2[2^n/3] + (2^n \bmod 3) - 1 \leq 2^n - (x_2 + x_3) + [x_3/2].$$

Доказательство. Пусть $2^n = 3q + r, q \geq 0, r \in [1, 2]$. Рассмотрим 4 случая в зависимости от значений, которые принимает x_2.

1) $x_2 = 0$. Поскольку $x_3 = (2^n - 2x_2)/3$, то такой случай невозможен в силу не равенства $2^n \bmod 3$ нулю.

2) $x_2 = 1$. Поскольку $x_3 = (2^n - 2x_2)/3$, тогда $r = 2$, а $x_3 = q$. Искомое неравенство принимает вид $2q + 1 \leq 3q + 2 - (1 + q) + [q/2] = 2q + 1 + [q/2]$, которое верно при любом $q \geq 0$.

3) $x_2 = 2$. Поскольку $x_3 = (2^n - 2x_2)/3$, тогда $r = 1$, а $x_3 = q - 1$. Искомое неравенство принимает вид $2q \leq 3q + 1 - (2 + q - 1) + [(q - 1)/2] = 2q + [(q - 1)/2]$, которое верно при любом $q - 1 \geq 0$, что и имеем, поскольку $x_3 = q - 1 \geq 4$.

4) $x_2 = 3$. Поскольку $x_3 = (2^n - 2x_2)/3$, то такой случай невозможен в силу не равенства $2^n \bmod 3$ нулю.

□

Пусть A — алгоритм расшифровки $F(n, k)$, покрывающий все 2^n наборы. Тогда через $C(A)$ будем обозначать такое число q, что первые $q-1$ запросов алгоритма A покрывают не все наборы, а q запросов покрывают все 2^n наборов. Представим, что каждый набор — это одноэлементное множество. Будем говорить, что запрос (a, b) объединяет два множества, которым принадлежат наборы a и b. Тогда под $N(A, x, y)$ будем понимать количество множеств мощности y, которые образовались после отправки первых x запросов алгоритма A.

Заметим, что для $k = 3$ можно немного уточнить приведенную в теореме 7 верхнюю оценку $2^n-(x_2+x_3)+[max(x_2, x_3)/2]$. Поэтому перейдем к доказательству следующей леммы.

Лемма 8. *Пусть $n \geq 4$ и для целых положительных x_2, x_3 верно равенство $2^n = 2x_2 + 3x_3$. Тогда справедлива следующая верхняя оценка*

$$\varphi_{CQ}(n, 3) \leq 2^n - (x_2 + x_3) + [max(x_2 - 1, x_3)/2].$$

Доказательство. Рассмотрим алгоритм A расшифровки функции, приведенный в теореме 7. Соответственно, зададим $(2^n - x_2 - x_3)$ запросов, образовав $N(A, 2^n - (x_2 + x_3), 3) = x_3$ множеств мощности 3 и $N(A, 2^n-(x_2+x_3), 2) = x_2$ множеств мощности 2 соответственно. Причем, сначала будем задавать запросы, относящиеся ко множествам мощности 3, а лишь затем ко множествам мощности 2.

При этом заметим что, если ответ на каждый из первых $(2^n-x_2-x_3)$ запросов был 0, тогда все единицы находятся в одном из x_3 множеств мощности 3 и для нахождения нужного множества мощности 3 надо дополнительно задать $[x_3/2]$ запросов, то есть суммарно будет задано $2^n - (x_2 + x_3) + [x_3/2]$ запросов. Иначе, если хоть раз был получен ответ отличный от 0, тогда стало известно значение на всех элементах множества 2 или множества 3.

- Если первый ответ отличный от 0 был получен на запросах, относящихся к формированию множеств мощности 2, тогда все множества мощности 3 уже сформированы, в них точно не находятся единицы, а значит единицы будут находиться в одном из x_2-1 множеств мощности 2. Соответственно, суммарно придется потратить $2^n - (x_2 + x_3) + [(x_2 - 1)/2]$ запросов.

- Если первый ответ отличный от 0 был получен на запросах, относящихся к формированию множеств мощности 3, тогда возможно три ситуации.

    1) Запрос объединял два одноэлементных множества. Соответственно, найдена только одна единица, и сформировано множество мощности 2. Будем считать, что мы отошли на один запрос от нашего плана сначала только формировать множества мощности 3, а затем множества мощности 2. Иными словами, мы с опережением сформировали одно множество мощности 2 и про него уже все знаем. Также нам известно, что

останется найти еще две единицы. Поэтому если мы запросим оставшиеся запросы, чтобы сформировать x_2 множеств мощности 2 и x_3 множеств мощности 3, и всегда в ответ получим 0, то две единицы будут содержаться в одном из $x_2 - 1$ множеств мощности 2. В этом случае, суммарно будет задано $2^n - (x_2 + x_3) + [(x_2 - 1)/2]$ запроса. Если в процессе формирования этих множеств мы еще раз получим ответ отличный от 0, то мы за $2^n - (x_2 + x_3)$ запросов найдем все единицы, дополнительные запросы не понадобятся.

2) Запрос объединял одно- и двухэлементное множество. По ответу на запрос мы однозначно поймем, какое из этих множеств состоит из единиц функции. Если они в двухэлементном множестве, тогда останется найти оставшуюся единицу и она раскроется в течение $2^n-(x_2+x_3)$ запросов при формировании x_2 множеств мощности 2 и x_3 множеств мощности 3, в силу этого дополнительные запросы не пригодятся и суммарно будет задано всего лишь $2^n - (x_2 + x_3)$ запросов.

   Если единица в одноэлементном множестве, тогда очевидно, что осталось найти две единицы и поэтому в принципе можно перестать формировать трехэлементные множества, а продолжить искать две единицы среди оставшихся непокрытых t наборов, где $2^n - 3x_3 \leq t \leq 2^n - 3$, алгоритмом, описанном в доказательстве леммы 1. Но для удобства вычисления числа запросов, доопросим все запросы, необходимые для формирования x_3 множеств мощности 3, и затем вместо формирования x_2 множеств мощности 2, воспользуемся алгоритмом, описанном при доказательстве верхней оценки леммы 1, то есть будем формировать множества мощности 3. Суммарно будет задано $2x_3$ запросов для формирования x_3 множеств мощности 3, а затем по алгоритму леммы 1 $2[(2^n-3x_3)/3]+((2^n-3x_3) \bmod 3)-1 = 2[2^n/3] - 2x_3 + (2^n \bmod 3) - 1$, что приводит нас к оценке $2x_3 + 2[2^n/3] - 2x_3 + (2^n \bmod 3) - 1 = 2[2^n/3] + (2^n \bmod 3) - 1$. Соответственно, в случае, когда запрос с ответом отличном от 0 объединял одно- и двухэлементное множество, ученик будет вынужден задать не более $max(2[2^n/3] + (2^n \bmod 3) - 1, 2^n - (x_2+x_3)+[(x_2-1)/2])$. Определим условия на x_2, x_3, при которых этот максимум был равен $2^n-(x_2+x_3)+[(x_2-1)/2]$. Тогда в целом при таком алгоритме расшифровки с учетом оценки $2^n - (x_2 + x_3) + [x_3/2]$ случая, описанного выше, понадобит-

ся не более $2^n - (x_2 + x_3) + [max(x_2 - 1, x_3)]/2$ запросов, что и докажет оценку данной леммы для ситуации, когда первый ответ отличный от 0 приходит во время первых $2x_3$ запросов. Перейдем доказательству неравенства $2[2^n/3] + (2^n \bmod 3) - 1 \leq 2^n - (x_2 + x_3) + [(x_2 - 1)/2] = x_2 + 2x_3 + [(x_2 - 1)/2]$. Если из обеих частей отбросим общие $2x_3$ запросы, останется лишь доказать неравенство $2[(2^n - 3x_3)/3] + ((2^n - 3x_3) \bmod 3) - 1 \leq x_2 + [(x_2 - 1)/2]$. В силу леммы 6, это неравенство имеет место при $x_2 \geq 4$.

Следовательно можно заключить, что необходимо задать в худшем случае $max(2[2^n/3]+(2^n \bmod 3)-1, 2^n-(x_2+x_3)+[max(x_2-1, x_3)/2])$. При этом, при $x_2 \geq 4$ число запросов равно $2^n-(x_2+x_3)+[max(x_2-1, x_3)/2]$, как и утверждается в данной лемме.

При $x_2 < 4$ число запросов равно $max(2[2^n/3] + (2^n \bmod 3) - 1, 2^n - (x_2 + x_3) + [x_3/2])$. Поскольку $n \geq 4$, то $(2^n - 2x_2) = 3x_3 \geq 10, x_3 \geq 4$, то есть $max(x_3, x_2 - 1) = x_3$. Осталось показать, что $2[2^n/3] + (2^n \bmod 3) - 1 \leq 2^n - (x_2 + x_3) + [x_3/2]$, а это было сделано в лемме 7.

Следовательно, оценка данной леммы доказана.

□

Будем говорить, что $x_2, x_3, \ldots, x_{2^n}$ *удовлетворяют условию* $\mu(n)$, если $x_2, x_3, \ldots, x_{2^n}$ — целые неотрицательные числа, а также выполнено

$$2x_2 + 3x_3 + 4x_4 + \ldots + 2^n x_{2^n} = 2^n, x_2 + x_3 > 1.$$

Рассмотрим следующую задачу. Пусть n — целое число, не меньшее 4. При условии, что $x_2, x_3, \ldots, x_{2^n}$ удовлетворяют условию $\mu(n)$, требуется максимизировать функцию

$$M(x_2, x_3, \ldots, x_{2^n}) = \begin{cases} x_2 + x_3 + x_4 + \ldots + x_{2^n} - [x_3/2], & \text{если } x_2 \leq x_3, \\ x_2 + x_3 + x_4 + \ldots + x_{2^n} - [(x_2 - 1)/2], & \text{если } x_2 > x_3. \end{cases}$$

Эту функцию можно переписать в следующем виде

$$M(x_2, x_3, \ldots, x_{2^n}) = x_2 + x_3 + x_4 + \ldots + x_{2^n} - [max(x_2 - 1, x_3)/2].$$

Лемма 9. *Пусть $a_2, a_3, \ldots, a_{2^n}$ удовлетворяют условию $\mu(n)$, помимо этого хотя бы для одного $t \in [4, 2^n]$ верно $a_t > 0$. Тогда существуют $b_2, b_3, \ldots, b_{2^n}$, для которых справедливо неравенство $M(a_2, a_3, \ldots, a_{2^n}) < M(b_2, b_3, \ldots, b_{2^n})$. При этом $b_2, b_3, \ldots, b_{2^n}$ удовлетворяют условию $\mu(n)$*

и отличаются от $a_2, a_3, \ldots, a_{2^n}$ *хотя бы в одном члене, то есть* $a_j \neq b_j$ *для некоторого* $j \in [2, 2^n]$.

Доказательство. Рассмотрим два случая в зависимости от четности t.

1) Пусть $t = 2 \cdot p$, p — целое. Тогда положим $b_2 = a_2 + p, b_3 = a_3, b_4 = a_4, \ldots, b_{t-1} = a_{t-1}, b_t = a_t - 1, b_{t+1} = a_{t+1}, \ldots, b_{2^n} = a_{2^n}$. Заметим, что $b_2, b_3, \ldots, b_{2^n}$ удовлетворяют $\mu(n)$ условию. Рассмотрим значение $M(b_2, b_3, \ldots, b_{2^n}) = b_2 + b_3 - [0.5 \cdot max(b_2 - 1, b_3)] + (b_4 + b_5 + \ldots + b_{2^n}) = M(a_2, a_3, \ldots, a_{2^n}) + p - 1 + [0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)]$. Осталось показать, что $p - 1 + [0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)] > 0$.

   Заметим, что $p \geq 2$, так как $t > 3$.

   Если $a_2 + p - 1 \leq a_3$, то $[0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)] = 0$.

   Если $a_2 - 1 > a_3$, то $[0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)] = [0.5 \cdot (a_2 + p - 1)] - [0.5 \cdot (a_2 - 1)] > 0$.

   Если $a_2 - 1 \leq a_3, a_2 + p - 1 \geq a_3$, то $[0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)] = [0.5 \cdot (a_2 + p - 1)] - [0.5 \cdot a_3] \geq 0$.

   Приходим к выводу, что $p - 1 + [0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)] > 0$.

2) Пусть $t = 2 \cdot p + 1$, p — целое. Заметим, что $t > 3$, поэтому $p \geq 2$. Тогда положим $b_2 = a_2 + (p-1), b_3 = a_3 + 1, b_4 = a_4, \ldots, b_{t-1} = a_{t-1}, b_t = a_t - 1, b_{t+1} = a_{t+1}, \ldots, b_{2^n} = a_{2^n}$. Заметим, что $b_2, b_3, \ldots, b_{2^n}$ удовлетворяют $\mu(n)$ условию.

   Рассмотрим значение $M(b_2, b_3, \ldots, b_{2^n}) = b_2 + b_3 - [0.5 \cdot max(b_2 - 1, b_3)] + (b_4 + b_5 + \ldots + b_{2^n}) = M(a_2, a_3, \ldots, a_{2^n}) + p - 1 + 1 - 1 + [0.5 \cdot max(a_2 + p - 2, a_3 + 1)] - [0.5 \cdot max(a_2 - 1, a_3)]$. Осталось показать, что $p - 1 + [0.5 \cdot max(a_2 + p - 2, a_3 + 1)] - [0.5 \cdot max(a_2 - 1, a_3)] > 0$.

   Если $a_2 + p - 2 \leq a_3 + 1$, следовательно, $a_2 - 1 \leq a_3$, то $[0.5 \cdot max(a_2 + p - 1, a_3 + 1)] - [0.5 \cdot max(a_2 - 1, a_3)] = [0.5 \cdot (a_3 + 1)] - [0.5 \cdot a_3] \geq 0$.

   Если $a_2 - 1 > a_3$, то $[0.5 \cdot max(a_2 + p - 2, a_3 + 1)] - [0.5 \cdot max(a_2 - 1, a_3)] = [0.5 \cdot (a_2 + p - 2)] - [0.5 \cdot (a_2 - 1)] \geq 0$.

   Если $a_2 - 1 \leq a_3, a_2 + p - 2 \geq a_3 + 1$, то $[0.5 \cdot max(a_2 + p - 1, a_3)] - [0.5 \cdot max(a_2 - 1, a_3)] = [0.5 \cdot (a_2 + p - 2)] - [0.5 \cdot a_3] \geq 0$.

   Приходим к выводу, что $p - 1 + [0.5 \cdot max(a_2 + p - 2, a_3 + 1)] - [0.5 \cdot max(a_2 - 1, a_3)] > 0$.

□

Из леммы 9 вытекает следующее следствие.

Следствие 1. *Пусть* $a_2, a_3, \ldots, a_{2^n}$ *удовлетворяют условию* $\mu(n)$*, причем* $a_4+a_5+\ldots+a_{2^n} > 0$*. Тогда* $M(a_2, a_3, a_4, \ldots, a_{2^n}) < M(a_2, a_3, 0, \ldots, 0)$.

Лемма 10. *Пусть* $n \geq 4$*,* $a_2, a_3, \ldots, a_{2^n}$ *удовлетворяют условию* $\mu(n)$ *и выполнено* $a_4 = a_5 = \ldots = a_{2^n} = 0, a_2 < a_3$*. Тогда существуют* $b_2, b_3, \ldots, b_{2^n}$*, для которых справедливо неравенство* $M(a_2, a_3, \ldots, a_{2^n}) \leq M(b_2, b_3, \ldots, b_{2^n})$*. При этом* $b_2, b_3, \ldots, b_{2^n}$ *удовлетворяют условию* $\mu(n)$ *и верно неравенство* $b_3 < a_3$.

Доказательство. Поскольку $a_3 + a_2 > 1, a_2 < a_3$, то либо $a_2 = 0, a_3 \geq 2$, либо $a_2 \geq 1, a_3 \geq 2$. Если $a_3 = 2$, а $a_2 \leq 1$, $3a_3 + 2a_2 \leq 8 < 2^n$ при $n \geq 4$. Следовательно, если $a_2 < a_3$, то $a_3 > 2$. Положим равными $b_2 = a_2 + 3, b_3 = a_3 - 2, b_4 = b_5 = \ldots = b_{2^n} = 0$. Очевидно, что $b_2, b_3, b_4, \ldots, b_{2^n}$ удовлетворяют условию $\mu(n)$. Рассмотрим значение выражения $M(b_2, b_3, b_4, \ldots, b_{2^n}) = b_2 + b_3 - [0.5 \cdot max(b_2 - 1, b_3)] + (b_4 + \ldots + b_{2^n}) = M(a_2, a_3, a_4, \ldots, a_{2^n}) + 1 + [0.5 \cdot max(a_2 + 3 - 1, a_3 - 2)] - [0.5 \cdot a_3]$. Осталось показать, что $1 + [0.5 \cdot max(a_2 + 2, a_3 - 2)] - [0.5 \cdot a_3] \geq 0$.

Если $a_2 + 2 \leq a_3 - 2$, то $1 + [0.5 \cdot max(a_2 + 2, a_3 - 2)] - [0.5 \cdot a_3] = 1 + [0.5 \cdot (a_3 - 2)] - [0.5 \cdot a_3] = 1 + [0.5 \cdot a_3] - 1 - [0.5 \cdot a_3] = 0$.

Если $a_2 + 2 > a_3$, то $1 + [0.5 \cdot max(a_2 + 2, a_3 - 2)] - [0.5 \cdot a_3] = 1 + [0.5 \cdot (a_2 + 2)] - [0.5 \cdot a_3] > 0$.

Если $a_3 - 2 \leq a_2 + 2 < a_3$, то $1 + [0.5 \cdot max(a_2 + 2, a_3 - 2)] - [0.5 \cdot a_3] = 1 + [0.5 \cdot (a_2 + 2)] - [0.5 \cdot a_3] \geq 1 + [0.5 \cdot a_3] - 1 - [0.5 \cdot a_3] = 0$. □

Из леммы 10 и следствия 1 вытекает следующее следствие.

Следствие 2. *Чтобы при* $n \geq 4$ *найти максимальное значение функции* $M(b_2, b_3, \ldots, b_{2^n})$*, достаточно перебрать удовлетворяющие условию* $\mu(n)$ $b_2, b_3, \ldots, b_{2^n}$*, для которых выполнены ограничения* $b_2 \geq b_3, b_4 = b_5 = \ldots = b_{2^n} = 0$.

Лемма 11. *Если* $n \geq 4$*, то максимальное значение функции* $M(x_2, x_3, \ldots, x_{2^n})$ *равно* $]3/2 \cdot [2^n/5][+[(2^n \bmod 5)/2]$ *и достигается при следующих* $x_2, x_3, \ldots, x_{2^n}$*:*

- $x_2 = [2^n/5]+2, x_3 = [2^n/5]-1, x_4 = \ldots = x_{2^n} = 0$*, если* $2^n \bmod 5 = 1$*,*
- $x_2 = [2^n/5] + 1, x_3 = [2^n/5], x_4 = \ldots = x_{2^n} = 0$*, если* $2^n \bmod 5 = 2$*,*

- $x_2 = [2^n/5]+3, x_3 = [2^n/5]-1, x_4 = \ldots = x_{2^n} = 0$, *если* $2^n \bmod 5 = 3$,
- $x_2 = [2^n/5] + 2, x_3 = [2^n/5], x_4 = \ldots = x_{2^n} = 0$, *если* $2^n \bmod 5 = 4$.

Доказательство. Согласно следствию 2, для поиска $x_2, x_3, \ldots, x_{2^n}$ достаточно перебрать $x_2, x_3, \ldots, x_{2^n}$, удовлетворяющие условию $\mu(n)$ и следующим ограничениям

1) $x_4 = x_5 = \ldots = x_{2^n} = 0$;

2) $x_2 \geq x_3$.

Отсюда следует, что $2^n = 2x_2 + 3x_3$, следовательно $2^n - 2x_2$ должен быть кратен 3.

1) Если $2^n \bmod 3 = 1$, то должно выполняться $x_2 \bmod 3 = 2$.

2) Если $2^n \bmod 3 = 2$, то должно выполняться $x_2 \bmod 3 = 1$.

Из равенства $2^n = 2x_2+3x_3$ следует то, что $x_2 \neq x_3$, а значит $x_2 > x_3$.

Соответственно, x_2 подберем в соответствии с этими требованиями. Учитывая, что $x_3 = (2^n - 2x_2)/3 < x_2$, получаем $x_2 \geq]2^n/5[$.

Рассмотрим значение функции $M(x_2, x_3, x_4, \ldots, x_{2^n}) = x_2 + x_3 - [0.5 \cdot max(x_2 - 1, x_3)] =](x_2+1)/2[+x_3 =](x_2+1)/2[+(2^n - 2x_2)/3$. Рассмотрим два случая в зависимости от четности x_2.

1) Если x_2 — нечетное, тогда $M(x_2, x_3, x_4, \ldots, x_{2^n}) = (x_2+1)/2+(2^n - 2x_2)/3 = (2^{n+1} - x_2 + 3)/6$. Учитывая ограничение $x_2 \geq]2^n/5[$, получаем, что максимум $M(x_2, x_3, x_4, \ldots, x_{2^n})$ достигается при x_2 наименьшем целом числе, неменьшим $]2^n/5[$ и дающий остаток от деления на 3 равный $3 - 2^n \bmod 3$.

2) Если x_2 — четное, тогда $M(x_2, x_3, x_4, \ldots, x_{2^n}) = x_2/2 + 1 + (2^n - 2x_2)/3 = (2^{n+1} - x_2 + 6)/6$. Учитывая ограничение $x_2 \geq]2^n/5[$, получаем, что максимум $M(x_2, x_3, x_4, \ldots, x_{2^n})$ достигается при x_2 наименьшем целом числе, неменьшим $]2^n/5[$ и дающий остаток от деления на 3 равный $3 - 2^n \bmod 3$.

Заметим, что если $n \bmod 4 = 0$, то $2^n \bmod 3 = 1, 2^n \bmod 5 = 1$. Если $n \bmod 4 = 1$, то $2^n \bmod 3 = 2, 2^n \bmod 5 = 2$. Если $n \bmod 4 = 2$, то $2^n \bmod 3 = 1, 2^n \bmod 5 = 4$. Если $n \bmod 4 = 3$, то $2^n \bmod 3 = 2, 2^n \bmod 5 = 3$.

Пусть $2^n = 3q+1 = 5t+r$, где q, t, r — целые неотрицательные числа, $r \in \{1, 4\}$. Соответственно, $]2^n/5[= t + 1$.

Рассмотрим два случая относительно остатков от деления 2^n на 5.

- $r = 1$. Заметим, что t — нечетное, так как $5t = 2^n - 1$. Теперь рассмотрим возможные остатки от деления t на 3. Если $t \bmod 3 = 1$, то $2^n = 5(3p+1)+1 = 3 \cdot 5p + 6 \neq 3q + 1$. Если $t \bmod 3 = 2$, то $2^n = 5(3p+2)+1 = 3 \cdot 5p + 11 \neq 3q + 1$. Если $t \bmod 3 = 0$, то $2^n = 5 \cdot 3p + 1 = 3 \cdot 5p + 1 = 3q + 1$. Отсюда следует, что $t \bmod 3 = 0$.

  Тогда $x_2 = t + 2, x_3 = (2^n - 2x_2)/3 = (3t + r - 4)/3 = t - 1$. Соответственно, $M(x_2, x_3, x_4, \ldots, x_{2^n}) = 2t + 1 - [(t + 2 - 1)/2] = 2t + 1 - (t + 1)/2 = (3t + 1)/2 =]3/2 \cdot [2^n/5][$.

- $r = 4$. Обратим наше внимание, что t — четное, так как $5t = 2^n - 4$. Теперь рассмотрим возможные остатки от деления t на 3. Если $t \bmod 3 = 1$, то $2^n = 5(3p+1)+4 = 3 \cdot 5p + 9 \neq 3q + 1$. Если $t \bmod 3 = 2$, то $2^n = 5(3p+2)+4 = 3 \cdot 5p + 14 \neq 3q + 1$. Если $t \bmod 3 = 0$, то $2^n = 5 \cdot 3p + 4 = 3q + 1$. Отсюда следует, что $t \bmod 3 = 0$.

  Тогда $x_2 = t+2, x_3 = (2^n - 2x_2)/3 = (3t+r-4)/3 = t$. Соответственно, $M(x_2, x_3, x_4, \ldots, x_{2^n}) = 2t + 2 - [(t + 2 - 1)/2] = 2t + 2 - t/2 = 3t/2 + 2 =]3/2 \cdot [2^n/5][+2$.

Пусть $2^n = 3q+2 = 5t+r$, где q, t, r — целые неотрицательные числа, $r \in \{2, 3\}$. Соответственно, $]2^n/5[= t + 1$.

Рассмотрим два случая относительно остатков на деление 2^n на 5.

- $r = 2$. Заметим, что t — четное, так как $5t = 2^n - 2$. Теперь рассмотрим возможные остатки от деления t на 3. Если $t \bmod 3 = 1$, то $2^n = 5(3p+1)+2 = 3 \cdot 5p + 7 \neq 3q + 2$. Если $t \bmod 3 = 2$, то $2^n = 5(3p+2)+2 = 3 \cdot 5p + 12 \neq 3q + 2$. Если $t \bmod 3 = 0$, то $2^n = 5 \cdot 3p + 2 = 3 \cdot 5p + 2 = 3q + 2$. Отсюда следует, что $t \bmod 3 = 0$.

  Тогда $x_2 = t+1, x_3 = (2^n - 2x_2)/3 = (3t+r-2)/3 = t$. Соответственно, $M(x_2, x_3, x_4, \ldots, x_{2^n}) = 2t + 1 - [(t + 1 - 1)/2] = 2t + 1 - t/2 = 3t/2 + 1 =]3/2 \cdot [2^n/5][+1$.

- $r = 3$. Также заметим, что t — нечетное, так как $5t = 2^n - 3$. Теперь рассмотрим возможные остатки от деления t на 3. Если $t \bmod 3 = 1$, то $2^n = 5(3p+1)+3 = 3 \cdot 5p + 8 = 3q + 2$. Если $t \bmod 3 = 2$, то $2^n = 5(3p+2)+3 = 3 \cdot 5p + 13 \neq 3q + 2$. Если $t \bmod 3 = 0$, то $2^n = 5 \cdot 3p + 3 \neq 3q + 2$. Отсюда следует, что $t \bmod 3 = 1$.

  Тогда $x_2 = t + 3, x_3 = (2^n - 2x_2)/3 = (3t + r - 6)/3 = t - 1$. Соответственно, $M(x_2, x_3, x_4, \ldots, x_{2^n}) = 2t + 2 - [(t + 3 - 1)/2] = 2t + 2 - (t + 1)/2 = (3t + 3)/2 =]3/2 \cdot [2^n/5][+1$.

□

Лемма 12. *Пусть $n \geq 7$, q, r — целые положительные числа из выражения $2^n = 5 \cdot q + r, r \in [1, 4]$. Тогда справедливо неравенство*

$$2^n - 1.5 \cdot q - [0.5 \cdot r] < 3 \cdot 2^{n-2} - 2.$$

Доказательство. Рассмотрим разность $2^n - 1.5 \cdot q - [0.5 \cdot r] - 3 \cdot 2^{n-2} + 2 = 2^{n-2} - 3/10 \cdot (2^n - r) + 2 - [0.5 \cdot r] = 1/10 \cdot (10 \cdot 2^{n-2} - 12 \cdot 2^{n-2} + 3r + 20 - 10 \cdot [0.5 \cdot r]) = 1/10 \cdot (-2^{n-1} + 3r - 10 \cdot [0.5 \cdot r] + 20)$. Последнее выражение отрицательно при $n \geq 7$ независимо от того, какое значение принимает r. □

Лемма 13. *При $n \geq 4$ справедлива следующая верхняя оценка*

$$\varphi_C(n, 3) \leq 2^n -]3/2 \cdot [2^n/5][-[(2^n \bmod 5)/2].$$

Доказательство. Подставим в оценку леммы 8 приводимые в лемме 11 значения x_2, x_3, а $x_4, \ldots, x_{2^n}$ положим равными 0. Заметим, что они удовлетворяют условию теоремы 7. Причем, получаемая верхняя оценка равна $2^n - M(x_2, x_3, 0, \ldots, 0)$. Согласно лемме 11, при выбранных таким образом x_2, x_3 достигается минимум $2^n -]3/2 \cdot [2^n/5][-[(2^n \bmod 5)/2]$. □

Лемма 14. *Если для расшифровки функции $f \in F(n, 3)$ ученик использует алгоритм, покрывающий не все наборы, тогда для однозначного восстановления функции ему потребуется как минимум $3 \cdot 2^{n-2} - 2$ запросов.*

Доказательство. Пусть $2^n = 4x + 4$, где x — целое. Пусть алгоритмом расшифровки не покрыто $r > 0$ наборов, при этом задано y запросов и ученик по ответам на свои запросы однозначно может восстановить загаданную функцию. Заметим, что $3x + 1 = 3 \cdot 2^{n-2} - 2$. В роли учителя на все первые $3x + 1$ запросов будем отвечать 0. Ответы на последующие запросы определим позднее при рассмотрении разных случаев.

Обратим внимание, что возможен один из двух случаев: либо все непокрытые наборы нули функции, либо все они являются ее единицами. Поскольку если бы часть непокрытых наборов была нулями, а оставшиеся непокрытые наборы единицами, то невозможно было без дополнительных запросов понять, какие из них и есть единицы. После y запросов $2^n - r$ наборов распались на классы эквивалентности, то есть подразбились на множества наборов, про которые известно, что значение функции на всех элементах одного множества одинаковое. Про все

множества мощности не меньшее четырех ученик сразу понял, что в них нет единиц функции. Рассмотрим всевозможные значения r.

1) $r \geq 4$. Учитель может спрятать все единицы функции среди этих четырех непокрытых наборов. В силу этого, ученику недостаточно y наборов для однозначного восстановления функции, соответственно такой случай невозможен.

2) $r = 3$. Если среди множеств имеется множество мощности 3, тогда ученик не поймет единицы лежат в этом множестве или среди непокрытых наборов. Следовательно, множеств мощности 3 не должно быть. Если среди множеств имеется множество мощности 2, тогда ученик поймет, что две единицы лежат в каком-то из множеств мощности 2, но не поймет какой из непокрытых наборов является третьей единицей. Соответственно, множеств мощности 2 также не должно быть. Приходим к выводу, что раз ученик однозначно восстановил функцию после y запросов, значит все множества имеют мощность не меньше 4, а непокрытые наборы и есть единицы функции. Согласно лемме 5, для объединения $4x + 1$ наборов во множества мощности не меньшей 4, необходимо не менее $3x + 1$ запросов, иными словами, $y \geq 3 \cdot 2^{n-2} - 2$.

3) $r = 2$. Если среди множеств имеется множество мощности 2, тогда ученик поймет, что две единицы лежат в каком-то из множеств мощности 2, но не поймет какой из непокрытых наборов является третьей единицей. Соответственно, множеств мощности 2 также не должно быть. Если среди множеств имеются хотя бы два множества мощности 3, тогда ученик не определит, в каком из этих множеств лежат единицы функции. Исходя из этого возможные следующие два случая.

   - Имеется ровно одно множество мощности 3 и несколько множество мощности не меньшей 4. На образование множества мощности 3 необходимо 2 запроса, а на покрытие оставшихся $4x - 1$ наборов, согласно лемме 5, понадобится $3(x - 1) + 3$ запроса. Отсюда следует, что $y \geq 3x + 2 \geq 3 \cdot 2^{n-2} - 2$.
   - Отсутствуют множества мощности 3 и все наборы разбились на множества мощности не меньшей 4. Согласно лемме 5, для этого необходимо $3x + 2$ запроса. На первые $3x + 1$ запросов учитель отвечает 0. Если же ученик задает $(3x + 2)$-й запрос

и учитель видит, что если ответ на этот запрос будет равен 0, то покрытыми станут $4x + 2$ набора и они распадутся на множества мощности не меньшей 4, тогда в этом случае учитель отвечает не 0, раскрывая единицы в той компоненте запроса, которая относится ко множеству меньшей мощности. Тем самым, учитель гарантирует, что не обманывал ученика и действительно его ответы соответствуют какой-то функции из $F(n, 3)$.

Рассмотреть ответ на $(3x+2)$-й запрос необходимо было лишь для последней цели, а так и этот случай демонстрирует, что ученику потребуется как минимум $3x + 1$ запросов, то есть $y \geq 3 \cdot 2^{n-2} - 2$.

4) $r = 1$. Если имеется и множество мощности 2, и множество мощности 3, тогда ученик не поймет, единицы функции лежат во множестве мощности 3 или во множестве мощности 2 и непокрытом наборе. Если имеется несколько множеств мощности 3, тогда ученик не поймет, в каком из них лежат единицы функции. Если имеются несколько множеств мощности 2, тогда ученик может понять, что одна единица — это непокрытый набор, но не сможет определить, в каком из множеств мощности 2 лежат оставшиеся единицы. Следовательно, возможны следующие три случая.

- Имеется ровно одно множество мощности 2, нет множеств мощности 3 и имеются несколько множество мощности не меньшей 4. На образование множества мощности 2 необходим 1 запрос, а на покрытие оставшихся $4x + 1$ наборов, согласно лемме 5, понадобится $3x + 1$ запросов. Отсюда следует, что $y \geq 3x + 2 \geq 3 \cdot 2^{n-2} - 2$.
- Имеется ровно одно множество мощности 3, нет множеств мощности 2 и имеются несколько множество мощности не меньшей 4. На образование множества мощности 3 необходимо 2 запроса, а на покрытие оставшихся $4x$ наборов, согласно лемме 3, понадобится $3x$ запросов. Отсюда следует, что $y \geq 3x + 2 \geq 3 \cdot 2^{n-2} - 2$.
- Отсутствуют множества мощности 2 и 3 и все наборы разбились на множества мощности не меньшей 4. Согласно лемме 5, для этого необходимо $3x + 3$ запроса. На первые $3x + 2$ запросов учитель отвечает 0. Если же ученик задает $(3x + 3)$-й

запрос и учитель видит, что если ответ на этот запрос будет равен 0, то покрытыми станут $4x + 3$ набора и они распадутся на множества мощности не меньшей 4, тогда в этом случае учитель отвечает не 0, раскрывая единицы в той компоненте запроса, которая относится ко множеству меньшей мощности. Тем самым, учитель гарантирует, что не обманывал ученика и действительно его ответы соответствуют какой-то функции из $F(n, 3)$. Рассмотреть ответ на $(3x + 3)$-й запрос необходимо было лишь для последней цели, а так и этот случай демонстрирует, что ученику потребуется как минимум $3x + 1$ запросов, то есть $y \geq 3 \cdot 2^{n-2} - 2$.

□

Лемма 15. *Пусть* $2^n = 5 \cdot q + r, r \in [1, 4]$, *и для расшифровки функции* $f \in F(n, 3)$ *ученик использует алгоритм расшифровки* A, *покрывающий все* 2^n *наборов. Тогда ученик задаст не менее* $3 \cdot 2^{n-2} - 2$ *запросов, если* $N(A, C(A), 2) + N(A, C(A), 3) \leq 1$.

Доказательство. Пусть $2^n = 4x + 4$. Рассмотрим три случая.

1) $N(A, C(A), 2) + N(A, C(A), 3) = 0$

   После $C(A)$ запросов образовались множества мощности строго больше 3. Согласно лемме 3, для этого потребуется не менее $3 \cdot 2^{n-2}$ запросов.

2) $N(A, C(A), 2) = 1, N(A, C(A), 3) = 0$

   После $C(A)$ запросов образовались множества мощности строго больше 3 и ровно одно множество мощности 2. Согласно лемме 5, для образования множеств мощности не меньшей 4 из $2^n - 2 = 4x + 2$ потребуется не менее $3x + 2 = 3 \cdot 2^{n-2} - 1$ запросов.

3) $N(A, C(A), 2) = 0, N(A, C(A), 3) = 1$

   После $C(A)$ запросов образовались множества мощности строго больше 3 и ровно одно множество мощности 3. Согласно лемме 5, для образования множеств мощности не меньшей 4 из $2^n - 3 = 4x + 1$ потребуется не менее $3x + 1 = 3 \cdot 2^{n-2} - 2$ запросов.

□

Лемма 16. *Пусть $n \geq 7$. Тогда верно неравенство*

$$\varphi_C(n,3) \geq 2^n -]3/2 \cdot [2^n/5][-[(2^n \bmod 5)/2].$$

Доказательство. Из лемм 12, 14 вытекает то, что для расшифровки функции с тремя единицами невыгодно использовать алгоритм расшифровки, непокрывающий все наборы. Поскольку верхняя оценка леммы 13 утверждает, что $\varphi_C(n,3) \leq 2^n -]3/2 \cdot [2^n/5][-[(2^n \bmod 5)/2] \leq 2^n - 1.5 \cdot [2^n/5] - [(2^n \bmod 5)/2]$.

Соответственно, чтобы задать как можно меньше запросов, ученик вынужден использовать алгоритм, покрывающий все наборы. Пусть первый запрос, после ответа на который окажутся покрытыми все 2^n наборов, имеет номер w. Пусть после w запросов исходные 2^n одноэлементных множеств — наборов объединятся в x_2 множеств мощности 2, x_3 множеств мощности 3, ..., x_{2^n} множеств мощности 2^n, иными словами верно соотношение $2^n = 2x_2 + 3x_3 + 4x_4 + \ldots + 2^n \cdot x_{2^n}$. На первые w запросов ученика (a,b) учитель будет отвечать следующим образом:

- На первые $w-1$ запросов учитель отвечает 0.
- Ответ на w-й запрос определяется следующим образом.

  Если $x_2 \leq x_3$, учитель отвечает 0. Если $x_2 > x_3$, учитель отвечает

  1) 1, если a — один из непокрытых наборов, покрываемых запросом (a,b),
  2) -1, в противном случае.

Из леммы 15 следует $x_2 + x_3 > 1$.

Спустя w запросов ученик понимает, что во всех $x_4 + x_5 + \ldots + x_{2^n}$ множествах мощности строго больше трех нет единиц.

Если на w-й запрос ученик в ответ получит 0, тогда он поймет, что все единицы находятся в одном из множеств мощности 3. Для определения нужного множества ученику потребуется $[x_3/2]$ запросов.

Если на w-й запрос ученик в ответ получит отличный от 0, тогда он найдет в точности одну единицу. Не нарушая общности будем считать, что на w-й запрос (a,b) ученик получил в ответ 1. Тогда он понимает, что a — единица, а множество, представителем которого является набор b, состоит полностью из нулей. Следовательно, ученику остается найти оставшиеся 2 единицы и они очевидно лежат в каком-то из множеств мощности 2. Для определения нужного множества ученику потребуется

- $[x_2/2]$ запросов, если и b был уже покрыт первыми $w-1$ запросами, иными словами, суммарно множество с представителем в b и набор a образуют множество мощности хотя бы 3,

- $[(x_2-1)/2]$, если и a, и b оба не были покрыты первыми $w-1$ запросами, а значит суммарно после w запросов образуют множество мощности 2, соответственно, про одно из x_2 множеств ученик полностью знает значение функции на каждом его элементе, поэтому остается искать среди меньшего числа множеств множество с двумя единицами.

Заметим, что $w = x_2 + 2x_3 + 3x_4 + \ldots + (i-1)x_i + \ldots + (2^n-1)x_{2^n} = 2^n - (x_2 + x_3 + \ldots + x_{2^n})$, поскольку для образования множества мощности i необходимо $i-1$ запросов. Соответственно, если $x_2 \leq x_3$ ученик задаст не менее $2^n - (x_2 + x_3 + \ldots + x_{2^n}) + [x_3/2]$, а если $x_2 > x_3$, то вынужден будет задать не менее $2^n - (x_2 + x_3 + \ldots + x_{2^n}) + [(x_2-1)/2]$ запросов.

Цель ученика подобрать такие $x_2, x_3, \ldots, x_{2^n}$, чтобы минимизировать это количество запросов, а значит максимизировать величину $(x_2 + x_3 + \ldots + x_{2^n}) - [x_3/2]$ при $x_2 \leq x_3$ и $(x_2 + x_3 + \ldots + x_{2^n}) - [(x_2-1)/2]$ при $x_2 > x_3$, соответственно максимизировать функцию $M(x_2, x_3, \ldots, x_{2^n}) = (x_2 + x_3 + \ldots + x_{2^n}) - [max(x_2 - 1, x_3)/2]$. Согласно лемме 11, максимальное значение этой функции равно $]3/2 \cdot [2^n/5][+[(2^n \bmod 5)/2]$, а значит ученик вынужден задать не менее $2^n -]3/2 \cdot [2^n/5][-[(2^n \bmod 5)/2]$.

□

Доказательство теоремы 6 следует из лемм 13 и 16.

8. Анализ результатов

Исходя из результатов, можно сделать вывод, что помимо классов $\{x\}, \{x, \overline{x}\}$, приводимых в [10], еще и класс функций фиксированного веса таков, что сложность его расшифровки запросами на сравнение строго меньше сложности расшифровки запросами на значение.

Также показано, что с точки зрения сложности расшифровки для класса функций фиксированного веса лучшими являются запросы на расширенную эквивалентность. Между тем использовать запросы на значение и запросы на эквивалентность для этого класса нецелесообразно в силу того, что сложность расшифровки этими типами запросов схожа с восстановлением всего вектора значений функций.

9. Благодарность

Автор выражает благодарность своему научному руководителю — д.ф-м.н., профессору Э. Э. Гасанову за постановку задачи и помощь в работе.

Список литературы

[1] D. Angluin, “Queries and Concept Learning”, *Machine Learning*, **2**:0885-6125 (1988), 319–342.

[2] D. Angluin, “Queries Revisited”, *Theoretical Computer Science*, **313**:2 (2001), 175–194.

[3] A. Bertoni, N. Cesa-Bianchi, G. Fiorino, “Efficient learning with equivalence queries of conjunctions of modulo functions”, *Information Processing Letters*, **56**:1 (1995), 15–17.

[4] W. Maass, G. Turan, “How Fast Can A Threshold Gate Learn?”, *Computational Learning Theory and Natural Learning Systems*, **1** (1990), 381–414.

[5] Вороненко А. А., Чистиков Д. В., *Проблемы теоретической кибернетики. Материалы XVI Международной конференции*, Издательство Нижегородского университета, 2011.

[6] Гасанов Э. Э., “Расшифровка линейных функций ранжирования”, *Материалы XI Международного семинара «Дискретная математика и ее приложения» (Москва, 18-23 июня 2012 г.)*, 2012, 332–334.

[7] Хегай С. И., “Расшифровка полиномиальных функций ранжирования”, *Интеллектуальные системы*, **19**:1 (2015), 213–230.

[8] А. А. Абдель Маджид, “О сложности восстановления частичного порядка”, *Интеллектуальные системы*, **20**:4 (2016), 5–10.

[9] Быстрыгова А. В., “Параметро-эффективная расшифровка булевых функций из замкнутых классов Поста”, *Дискретная математика*, **31**:2 (2019), 34–58.

[10] Быстрыгова А. В., “Запросы на сравнение в задаче параметро-эффективной расшифровки булевых функций”, *Интеллектуальные системы*, **23**:4 (2019), 115–124.

[11] Калачев Г.В., “Оценки мощности плоских схем, реализующих функции с ограниченным числом единиц”, *Интеллектуальные системы*, **21**:1 (2017), 1–51.

Learning of Boolean fixed-weight functions

Bistrigova A.V.

This paper is concerned with the learning complexity of Boolean fixed-weight functions using membership queries, comparation queries, equivalence queries, and extended equivalence queries. Moreover, it is allowed to use only one type of queries during learning. This paper

gives exact values of learning complexity for all the types of queries besides comparation queries. The paper presents the upper bound on learning complexity for comparation queries. In addition, the paper demonstrates that the upper bound is equal to the lower bound for the 1-, 2-, 3-weight functions.

Keywords: Boolean fixed-weight functions, membership queries, comparation queries, equivalence queries, extended equivalence queries, exact learning.