Конкурсы и олимпиады

Тест Клепцына: с компьютером и без него

К. А. Кноп Л. Э. Медников

История вопроса. В осенний тур 30-го Турнира городов была включена следующая задача Виктора Клепцына.

Тест состоит из 30 *вопросов, на каждый есть* 2 *варианта ответа* (*один верный, другой нет*). *За одну попытку Витя отвечает на все вопросы, после чего ему сообщают, на сколько вопросов он ответил верно. Сможет ли Витя действовать так, чтобы гарантированно узнать все верные ответы не позже, чем после* 24*-й попытки*? (*Изначально Витя не знает ни одного ответа, тест всегда один и тот же.*)

ЛМ. Поначалу я не мог разгадать тест даже за 29 проб. Тем временем, местные[1)] любители Турнира городов сумели уменьшить число попыток до 21. Ознакомившись с их алгоритмами, я построил свой алгоритм разгадывания теста за 22 вопроса. Но конкуренция с молодёжью на этом пути была обречена: мне уже сообщили, что достаточно 18 попыток.

КК. Я заинтересовался этой задачей независимо от ЛМ и поначалу рассматривал её как задачу о взвешиваниях. Переформулировка исходной задачи на языке взвешиваний очевидна: «У Вити имеются 30 монет двух весов (известно, каких именно). За одно взвешивание (на одночашечных весах со стрелкой) можно узнать суммарный вес любого подмножества монет. За какое (как можно меньшее) число взвешиваний Витя сумеет гарантированно определить вес каждой монеты?» Родство двух этих задач легко объяснить. В исходной задаче о тесте в какой-то из попыток нужно дать все одинаковые ответы (например, ответы «да»[2)]). Из этой попытки мы узнаём количество ответов «да»

[1)] Второй автор живёт в Израиле.

[2)] Для единообразия мы считаем, что вариантами ответа на каждый вопрос являются «да» и «нет».

Математическое просвещение, сер. 3, вып. 14, 2010(204–213)

среди правильных ответов. Пусть их число равно k. В формулировке с монетами первой попыткой нужно взвесить все монеты, и мы узнаем количество монет каждого веса (если известные веса монет равны x и y, то по результату взвешивания $V = kx + (30 - k)y$ легко найти k). На каждой следующей попытке угадывания часть ответов «да» заменяется на «нет», а в ответ мы получаем информацию о том, сколько ответов угаданы в этой попытке. Если мы заменили p ответов, а число верно угаданных изменилось на q, это значит, что $(p+q)/2$ ответов из изменённых угаданы правильно, а остальные $(p-q)/2$ — ошибочны. Аналогично в задаче про взвешивания: если взвешены t монет, из которых p монет веса x, то общий вес равен $V = px + (t-p)y = ty + p(x-y)$. Зная этот вес, а также x, y и t, Витя немедленно определяет p: $p = \dfrac{V - ty}{x - y}$. Цель Вити — найти монеты веса x, что эквивалентно нахождению всех верных ответов на тест.

Задача об «опознании» монет двух весов не нова: её частный случай для 4 монет и 3 взвешиваний предлагался ещё 20 лет назад на Московской математической олимпиаде (8 класс, задача 4, автор Сергей Гашков, см. [2]). Впрочем, ещё на 5 лет раньше случай для 7 монет и 5 взвешиваний был опубликован как задача E3023 в American Mathematical Monthly. Правда, в задаче AMM веса монет были неизвестны, что не слишком усложняет ситуацию. Принципиальное же усложнение «теста Клепцына» по сравнению с его предшественниками — переход от маленьких чисел (4 и 7) к сравнительно большому (30).

ЛМ. При каждой следующей попытке Витя может использовать информацию, добытую в предыдущих попытках (такие алгоритмы называются *адаптивными*). Однако именно это условие плохо переводилось на понятные мне языки, и для начала я решил ограничиться *неадаптивным* алгоритмом: считать, что для разгадывания теста за n попыток Витя одновременно засылает на экзамен n своих агентов, которые вводят разные ответы согласно выданным им инструкциям. На основании полученных результатов Витя и должен определить верные ответы на все вопросы. В такой постановке задача легко перевелась на язык линейной алгебры.

Разрешимые системы векторов

Ответ каждого агента можно рассматривать как строку из ± 1 (единица — выбор ответа «нет», минус единица — ответа «да»)[3]. Неизвестный верный ответ — такая же строка. В качестве результата попытки удобно рассматривать не число верных ответов, а скалярное произведение этих строк — разность между числом верных и неверных ответов. (Эта разность и число верных ответов связаны простым линейным соотношением.)

[3] На первый взгляд, естественнее рассматривать строку из 1 и 0, но, как будет видно впоследствии, «симметричное» представление полезнее.

Ответы всех n агентов на фиксированный вопрос теста задают n-мерный вектор с координатами ± 1 (такие векторы будем называть *знаковыми*). *Знаковой суммой* набора векторов $u_1, \dots, u_k$ назовём любую сумму вида $\varepsilon_1 u_1 + \dots + \varepsilon_k u_k$, где каждый из коэффициентов ε_i равен 1 или -1.

Набор знаковых векторов $u_1, \dots, u_k$ назовём *разрешимым*, если по каждой их знаковой сумме можно однозначно восстановить знаки коэффициентов ε_i. (Это значит, что все 2^k знаковых сумм различны.)

Итак, неадаптивный вариант задачи сведён к такому: при каких n в n-мерном пространстве существует разрешимый набор длины 30 (из 30 векторов).

ЛМ. Сам по себе перевод на алгебраический язык ничего не даёт. Разумеется, разрешимым набором будет любой базис 30-мерного пространства, но зачем нам разгадывать тест за 30 попыток? (Тем более, что базис из 30 знаковых векторов ещё надо построить.) Однако переход от подбора строчек к подбору столбцов привёл к идее работать с 16-мерным пространством и использовать тот факт, что 16 — число вершин 4-мерного куба. На этом пути, как мне показалось, удалось построить разрешимый набор длины 29. Отсюда следовало бы, что в 17-мерном пространстве существует разрешимый набор длины 30, а это лучше, чем известный мне «рекорд».

КК. По моей просьбе Лео Брухис из Нью-Йорка запрограммировал перебор тестов на компьютере. Однако вскоре выяснилось, что прямой перебор за разумное время не проходит, и в этот момент очень кстати оказались идеи ЛМ. Получив от него «решение» набор из 17 тестов длины 29, Лео прогнал его на компьютере, и тут же обнаружил в нём ошибку. Затем он сумел запрограммировать «целенаправленный» подбор векторов, и в результате машина выдала разрешимый набор длины 33 — на три больше, чем было «нужно».

ЛМ. Внимательно посмотрев на результат компьютерного моделирования, я наконец-то пришёл к более осмысленному пониманию своей собственной идеи, и в результате придал ей следующий вид.

Лемма. Если для набора векторов $u_1, \dots, u_k$ найдётся такой набор линейных функционалов $\varphi_1, \dots, \varphi_{k-1}$, что

$$\varphi_j(u_j) > |\varphi_j(u_{j+1})| + |\varphi_j(u_{j+2})| + \dots + |\varphi_j(u_k)| \quad \text{для каждого } j = 1, \dots, k-1, \tag{$*$}$$

то набор $u_1, \dots, u_k$ разрешим.

Доказательство. Пусть $u = \varepsilon_1 u_1 + \dots + \varepsilon_k u_k$ — некоторая знаковая сумма. Тогда

$$\varphi_1(u) = \varepsilon_1 \varphi_1(u_1) + \varepsilon_2 \varphi_1(u_2) + \dots + \varepsilon_k \varphi_1(u_k).$$

По условию, $\varphi_1(u) > 0$, если $\varepsilon_1 = 1$, и $\varphi_1(u) < 0$, если $\varepsilon_1 = -1$. Это значит, что $\varepsilon_1 = \operatorname{sign} \varphi_1(u)$. Аналогично $\varepsilon_2 = \operatorname{sign} \varphi_2(u - \varepsilon_1 u_1)$, $\varepsilon_3 =$

$= \operatorname{sign}\varphi_3(u - \varepsilon_1 u_1 - \varepsilon_2 u_2)$, и т. д. Наконец, ε_k определяется из сравнения u_k с выражением $u - \varepsilon_1 u_1 - \dots - \varepsilon_{k-1} u_{k-1}$. $\square$

Заметим, что предъявление конкретных функционалов позволяет не только доказать абстрактную разрешимость набора, но и даёт эффективный алгоритм определения правильных ответов теста.

Осталось только построить наборы векторов и функционалов нужной длины.

Построение разрешимого набора

Мы будем строить достаточно длинный разрешимый набор в пространстве размерности $n = 2^m$. Поэтому вместо n-мерных векторов будем рассматривать функции $f(x_1, \dots, x_m)$ от m переменных, принимающих значения ± 1 (область определения такой функции состоит из $2^m = n$ точек $(\pm 1, \dots, \pm 1)$, а значения образуют n-мерный вектор).

Знаковому вектору соответствует функция, принимающая только значения ± 1 (такую функцию мы тоже будем называть знаковой).

Простой пример линейного функционала — сумма значений функции в нескольких точках. Собственно, нам понадобятся только такие функционалы.

В частности, обозначим через $\sigma(f)$ сумму значений функции f во всех 2^m точках области определения.

Заметим, что $\sigma(f) = 0$ для каждой функции, нечётной хотя бы по одной из переменных.

С другой стороны, для знаковой функции f число $\sigma(f)$ чётно и (по модулю) не превосходит 2^m.

Пусть A — k-элементное подмножество множества $\{1, 2, \dots, m\}$.

Обозначим через $\sigma_A(f)$ сумму значений функции f в 2^{m-k} точках, у которых все координаты с номерами из A равны 1.

Например, $\sigma_1(f)$ — сумма значений $f(1, x_2, \dots, x_m)$; $\sigma_{2,3}(f)$ — сумма значений $f(x_1, 1, 1, x_4, \dots, x_m)$ и т. д.

Заметим, что если f нечётна по одной из переменных, номер которой не принадлежит A, то $\sigma_A(f) = 0$.

С другой стороны, для любого чётного числа c от -2^{m-k} до 2^{m-k} найдётся нечётная по всем переменным с номерами из A знаковая функция f, для которой $\sigma_A(f) = c$.

Действительно, если, например, $A = \{1, \dots, k\}$, знаки чисел $f(1, \dots, 1, x_{k+1}, \dots, x_m)$ мы можем задать произвольно, добившись тем самым нужного значения их суммы, а остальные значения функции f однозначно восстанавливаются «по нечётности».

Теорема 1. Имеется разрешимый набор знаковых функций от m переменных длины

$$m+(m-1)\binom{m}{1}+(m-2)\binom{m}{2}+\dots+2\binom{m}{m-2}+\binom{m}{m-1}+1=m\cdot 2^{m-1}+1.$$

Доказательство. Рассмотрим такие знаковые функции $f^1, \ldots, f^m$, что $\sigma(f^i)=2^i$. Поставим каждой из этих функций в соответствие функционал σ и расположим их в порядке $f^m, \ldots, f^1$. Для каждого k от 1 до $m-1$ и каждого k-элементного подмножества A множества $\{1,\ldots,m\}$ рассмотрим *нечётные по всем переменным с номерами из* A знаковые функции $f^1_A, \ldots, f^{m-k}_A$, для которых $\sigma_A(f^i_A)=2^i$. (Выше показано, что такие функции существуют.) Поставим каждой из этих функций в соответствие функционал σ_A и расположим их в порядке $f^{m-k}_A, \ldots, f^1_A$. Наконец, рассмотрим нечётную по всем переменным функцию $f_{1,2,\ldots,m}$.

Указанный набор функций и функционалов, расположенных в порядке возрастания k, (функции, соответствующие множествам из равного количества элементов, можно ставить в любом порядке) удовлетворяет условиям леммы и, следовательно, разрешим.

Действительно, если A не содержит B (в частности, если $|A|\leqslant|B|$), то B содержит элемент i, не входящий в A, т. е. $\sigma_A(f^i_B)=0$.

С другой стороны,

$$\sigma_A(f^i_A)=2^i>2^{i-1}+\dots+2^1=\sigma_A(f^{i-1}_A)+\dots+\sigma_A(f^1_A). \qquad \square$$

ЛМ. Таким образом, за $16=2^4$ попыток Витя может разгадать тест не только из 30, но даже из $33=4\cdot2^3+1$ вопросов. Возникло естественное желание уменьшить число попыток до 15, а также научиться строить достаточно длинные разрешимые системы в пространствах любой размерности. Ещё через несколько дней я понял, что последняя задача сводится к теореме 1.

Теорема о разрешимых системах

Пусть $S_2(n)$ — сумма цифр в двоичной записи числа n, а $B(n)=\sum_{i=1}^{n-1}S_2(i)$. В частности, $B(2^m)=m\cdot2^{m-1}$. [4)]

Теорема 2. В n-мерном пространстве имеется разрешимый набор длины $B(n)+1$.

Вместо полного доказательства мы покажем, как из построенного в теореме 1 разрешимого набора длины 13 в 8-мерном пространстве получить

[4)] Последовательность $B(n)$ в онлайн-энциклопедии целочисленных последовательностей [1] имеет номер A000788: 0, 1, 2, 4, 5, 7, 9, 12, 13, 15, 17, 20, 22, 25, 28, 32, 33, 35, 37, 40, 42, 45, 48, 52, 54, 57, 60… .

разрешимые наборы в 7-, 6- и 5-мерном пространствах (длины соответственно 10, 8, 6). Действия в общем случае совершенно аналогичны.

Построенный набор длины $13 = B(8)+1$ состоял из функций (8-мерных векторов)

$$f^3,\ f^2,\ f^1,\ f_1^2,\ f_1^1,\ f_2^2,\ f_2^1,\ f_3^2,\ f_3^1,\ f_{1,2},\ f_{1,3},\ f_{2,3},\ f_{1,2,3}$$

и соответствующих функционалов

$$\sigma,\ \sigma,\ \sigma,\ \sigma_1,\ \sigma_1,\ \sigma_2,\ \sigma_2,\ \sigma_3,\ \sigma_3,\ \sigma_{1,2},\ \sigma_{1,3},\ \sigma_{2,3}.$$

Значение функции в точке $(-1,-1,-1)$ было нужно только для вычисления функционала σ. Остальные функционалы можно рассматривать на 7-мерном пространстве функций, определённых на оставшихся 7 точках. Ограничив все наши функции (кроме первых трёх) на эти 7 точек, мы получим разрешимый набор 7-мерных векторов g_1^2, g_1^1, g_2^2, g_2^1, g_3^2, g_3^1, $g_{1,2}$, $g_{1,3}$, $g_{2,3}$, $g_{1,2,3}$ длины 10 с функционалами σ_1, σ_1, σ_2, σ_2, σ_3, σ_3, $\sigma_{1,2}$, $\sigma_{1,3}$, $\sigma_{2,3}$. При этом длина набора уменьшится на $3 = S_2(7)$. Поэтому число функций с $B(8)+1$ уменьшилось до $B(7)+1$.

Далее заметим, что (из оставшихся функционалов) только при вычислении σ_1 нужно знать значение функции в точке $(1,-1,-1)$. Это позволяет ограничить значения всех функций, кроме g_1^2, g_1^1 на множество из шести оставшихся точек, то есть получить 6-мерный набор h_2^2, h_2^1, h_3^2, h_3^1, $h_{1,2}$, $h_{1,3}$, $h_{2,3}$, $h_{1,2,3}$ длины 8 с функционалами σ_2, σ_2, σ_3, σ_3, $\sigma_{1,2}$, $\sigma_{1,3}$, $\sigma_{2,3}$. (Длина набора уменьшилась ещё на $2 = S_2(6)$.)

Наконец заметим, что только при вычислении σ_2 нужно знать значение функции в точке $(-1,1,-1)$. Это позволяет ограничить значения всех функций, кроме h_2^2, h_2^1, на множество из пяти оставшихся точек и получить 5-мерный набор p_3^2, p_3^1, $p_{1,2}$, $p_{1,3}$, $p_{2,3}$, $p_{1,2,3}$ длины 6 с функционалами σ_3, σ_3, $\sigma_{1,2}$, $\sigma_{1,3}$, $\sigma_{2,3}$.

(Следующий шаг — исключение точки $(1,1,-1)$и функционала $\sigma_{1,2}$ приведёт к конструкции, изложенной в теореме 1 для $n=4$, $m=2$: σ_3 превратится в σ, а $\sigma_{1,3}$ и $\sigma_{2,3}$ — в σ_1 и $\sigma 2$.)

ЛМ. Увы, для $n = 15$ теорема 2 даёт разрешимый набор длины 29 (но не 30).

Проблема 1. Является ли построенный в теореме 2 разрешимый n-мерный набор максимальным в следующих смыслах:

а) к нему нельзя добавить ни одного знакового вектора без нарушения разрешимости;

б) не существует разрешимого n-мерного набора большей длины?

Хотелось бы, чтобы ответ был положительным, хотя для больших n в это довольно трудно поверить (хотя бы в случае б)). Но при $n = 4, 5$ ответ положительный (см. ниже упражнения 3 и 4 и задачу 2). Остаётся надежда получить ответ с помощью машинного эксперимента для ещё нескольких «малых» значений n.

Упражнение 1. Набор знаковых векторов является разрешимым тогда и только тогда, когда каждая знаковая комбинация каждого непустого поднабора отлична от нуля.[5)]

Упражнение 2. К разрешимому набору можно добавить любой знаковый вектор, не являющийся результатом знаковой комбинации какого-то поднабора.

Проблема 2. Можно ли уменьшить число попыток для разгадывания теста, используя адаптивный алгоритм?

> *КК.* Анатолий Воробей сообщил мне (со ссылкой на Томека Чайку, сотрудника Google) следующую информацию о машинном моделировании «теста Клепцына»: программа, написанная Т. Чайкой, пробует несколько случайно выбранных ответов на тест, для каждого из них подсчитывает, сколько вариантов приводят к каждому возможному ответу, и находит максимальное число вариантов — т. е. худший случай. Из всех ответов выбирается тот, у которого худший случай наименьший. Далее программа рекурсивно решает каждую из 31 подзадач, отвечающих возможным результатам выбранного ответа на тест, и возвращает максимум по подзадачам плюс 1. При 50 случайных вопросах ей удаётся решить задачу за 15 попыток, и это занимает несколько часов компьютерного счета.
>
> После того, как ЛМ уже построил разрешимые наборы для произвольного n, я стал просматривать библиографию, приведённую на сайте «Справочника целочисленных последовательностей», и наткнулся на статью Бернта Линдстрома [3], в которой решалась фактически та же задача построения разрешимого набора (там он назывался *detecting set*). Решение Линдстрома давало набор той же длины, что и решение ЛМ, при этом опиралось на построение такого набора натуральных чисел, для которого все подмножества имеют различные суммы. Однако Линдстром решал свою задачу «жадным» способом, предварительно сведя её к другой хорошо известной задаче — поиску $(0, 1)$-матриц с наибольшим значением определителя (Hadamard maximal determinant problem, см. последовательность A003432 в [1]). А именно, построив detecting set из n векторов с некоторым значением определителя, он затем переходил к $n + 1$ и искал такой вектор, дополняющий ранее построенный набор, который сделает максимально возможным значение следующего определителя. Кроме того, Линдстром аккуратно исследовал асимптотику и доказал, что построенная им конструкция при больших n не может являться оптимальной. Видимо, это же можно сказать и про решение ЛМ, так что ответ на проблему 1б, скорее всего, отрицателен.

[5)] Некий аналог линейной независимости.

Индуктивный подход

КК. Внимательно проанализировав решения, найденные участниками турнира и другими энтузиастами (особенно следует отметить огромный вклад Алексея Гладких), я вычленил из них идею «двух стадий»: сначала (первая стадия) показывается, какие именно величины достаточно знать, чтобы суметь решить тест для маленькой группы вопросов, а потом (вторая стадия) на всем тесте проводятся такие попытки, которые позволяют узнать нужные величины. Эта идея в конечном итоге привела к следующему построению.

Будем называть *каноническим* такой неадаптивный способ решения теста, при котором одна из попыток (для определённости, первая) представляет собой ответы «да» на все вопросы.

Лемма. Если существует канонический способ решить тест длины M за N попыток, то существует и канонический способ решить тест длины $2M + N - 1$ за $2N$ попыток.

Доказательство. Назовём *весом* множества вопросов X количество правильных ответов «да» на вопросы из множества X. *Взвешиванием* множества вопросов X будем называть такую попытку ответа на тест: на все вопросы множества X даётся ответ «да», а на все остальные вопросы — ответ «нет».

Отметим, что для канонических тестов результат каждого взвешивания в совокупности с результатом первой попытки позволяет вычислить вес множества X и его дополнения.

Разобьём $2M+N-1$ вопросов теста на три части, две из которых (назовём их A и B) содержат M вопросов, а третья часть C — остальные $N - 1$ вопросов. Заметим, что множества вопросов A, B и C не пересекаются.

По условию леммы правильные ответы на вопросы каждой из частей A или B можно узнать каноническим тестом из N попыток. Подмножества вопросов, которые мы бы взвешивали в этих попытках, мы будем обозначать $A_1, \ldots, A_{N-1}$ для части A, для части B взвешиваемые в каноническом тесте множества обозначим через $B_1, \ldots, B_{N-1}$.

Список попыток тоже разобьём на три части: первая попытка, вторая попытка, части «Sum» и «Diff» (из $N - 1$ попыток каждая). В первой попытке мы отвечаем «да» на все вопросы (это канонический тест). Во второй — взвешиваем множество B. В части «Sum» взвешиваем в I-й попытке $A_I \cup B_I \cup C_I$. В части «Diff» в I-й попытке взвешиваем $A_I \cup (B \setminus B_I)$. Рассмотрим, какие выводы можно сделать, получив результаты первых двух попыток и I-х попыток из «Sum» и «Diff», то есть узнав веса $A_I + B_I + C_I$, $A_I + B - B_I$ и B. Очевидно, что чётность суммы этих результатов совпадает с чётностью значения C_I, что даёт нам знание правильного ответа на вопрос C_I. Дальше из системы уравнений легко находим веса A_I и B_I.

Наконец, зная общее число ответов «да» во всем тесте, а также в частях B и C, мы можем определить вес A. Таким образом, по результатам проделанных попыток мы знаем результаты взвешиваний всех подмножеств $A_1, \dots, A_{N-1}$, A, всех подмножеств $B_1, \dots, B_{N-1}$, B и правильные ответы на все вопросы множества C. В силу выбора множеств A_I, B_I мы в состоянии по этим данным узнать правильные ответы все вопросы множеств A и B, на этом доказательство леммы закончено. $\square$

Индуктивный алгоритм построения решения задачи получается следующим образом: пользуясь очевидным результатом «2 вопроса за 2 попытки» и леммой для $N = 2$, найдем ответы для теста из $5 = 2 \cdot 2 + 1$ вопросов за 4 попытки. Пользуясь результатом «5 за 4» и леммой для $N = 4$, решим тест из $13 = 2 \cdot 5 + 3$ вопросов за 8 попыток. Наконец, воспользуемся результатом «13 за 8» и леммой для $N = 8$. В результате получим решение теста на $33 = 2 \cdot 13 + 7$ вопроса за 16 попыток. Отметим, что этот результат идеально соответствует теореме о разрешимом наборе: тест длины $m \cdot 2^{m-1} + 1$ разгадывается за 2^m попыток.

Ещё одна связь с гиперкубом

Упражнение 3. В n-мерном пространстве существует разрешимый набор длины k тогда и только тогда, когда можно отметить k вершин $(n-1)$-мерного куба так, что каждые два (несовпадающих) набора из равного числа отмеченных вершин имеют разные центры масс.

Упражнение 4. Если в 3-мерном кубе отметить 6 вершин, то четыре из них являются вершинами параллелограмма.

(Это значит, что в 4-мерном пространстве не существует разрешимого набора длины 6.)

Задача 1. Если в 4-мерном кубе отметить 8 вершин, то четыре из них являются вершинами параллелограмма. Для 7 вершин это неверно.

Задача 2. Если в 4-мерном кубе отметить 7 вершин, то найдутся два треугольника с общим центром масс, у которых все вершины отмечены. (Треугольники могут пересекаться.) Для 6 вершин это неверно.

(Это значит, что в 5-мерном пространстве не существует разрешимого набора длины 7.)

Проблема 3. При каком числе вершин выполнены утверждения задач 1 и 2 в 5-мерном кубе?

Проблема 4. В 5-мерном кубе отметили 9 вершин. Всегда ли найдутся два тетраэдра (возможно, вырожденных) с общим центром масс, у которых все вершины отмечены?

Для 8 вершин есть контрпример (это следует из утверждения 3).

Нижние границы

Итак, в конечном счёте мы нашли (даже двумя разными способами) решение задачи за 16 попыток. Насколько это близко к идеалу? Более общо, каковы нижние оценки для числа попыток, необходимых для решения теста длины 30?

Ниже — один из возможных подходов к проблеме нижних границ.

Во-первых, если мы в проделанных попытках поменяем на противоположные все свои ответы на один и тот же вопрос, то все результаты этих попыток изменятся на 1 — увеличатся, если мы поменяли неправильный ответ на правильный, и уменьшатся в противном случае. Поэтому если какой-то алгоритм позволяет узнать правильные ответы на все вопросы, то можно поменять в нем ответы так, чтобы в первой его попытке на все вопросы отвечать «да», — в результате тоже должен получиться работающий алгоритм.

Пусть ответ на первую попытку был равен 15. Это означает, что число вариантов уменьшилось с 2^{30} до $\binom{30}{15} = 155\,117\,520$. Если в любой из следующих попыток мы дали чётное число ответов «нет», то ответом будет нечётное число от 1 до 29 (15 возможных ответов), а если количество ответов «нет» чётно, то ответом будет чётное число от 0 до 30 (16 ответов). Итого в каждом случае число вариантов может сократиться не более чем в 16 раз, поэтому после 6 следующих попыток может остаться не менее чем $\binom{30}{15}/16^6$ вариантов заполнения теста. Целая часть этого числа больше 1 — она равна 8. Таким образом, 7 попыток не хватит (даже при адаптивном алгоритме).

Обобщение этого подхода даёт результат $1 + \log_K \binom{N}{K}$, где $K = N/2$.

Список литературы

[1] http://www.research.att.com/~njas/sequences/

[2] http://kvant.mirror1.mccme.ru/1988/09/p72.htm

[3] B. Lindstrom. *On a combinatorial problem in number theory* // Canad. Math. Bull. Vol. 8, 1965. P. 477–490.

К. А. Кноп, ЮМШ при СПбГУ
E-mail: kostyaknop@gmail.com
Сайт: http://www.kknop.com

Л. Э. Медников, Хайфа, Израиль
E-mail: lmednikov@mail.ru