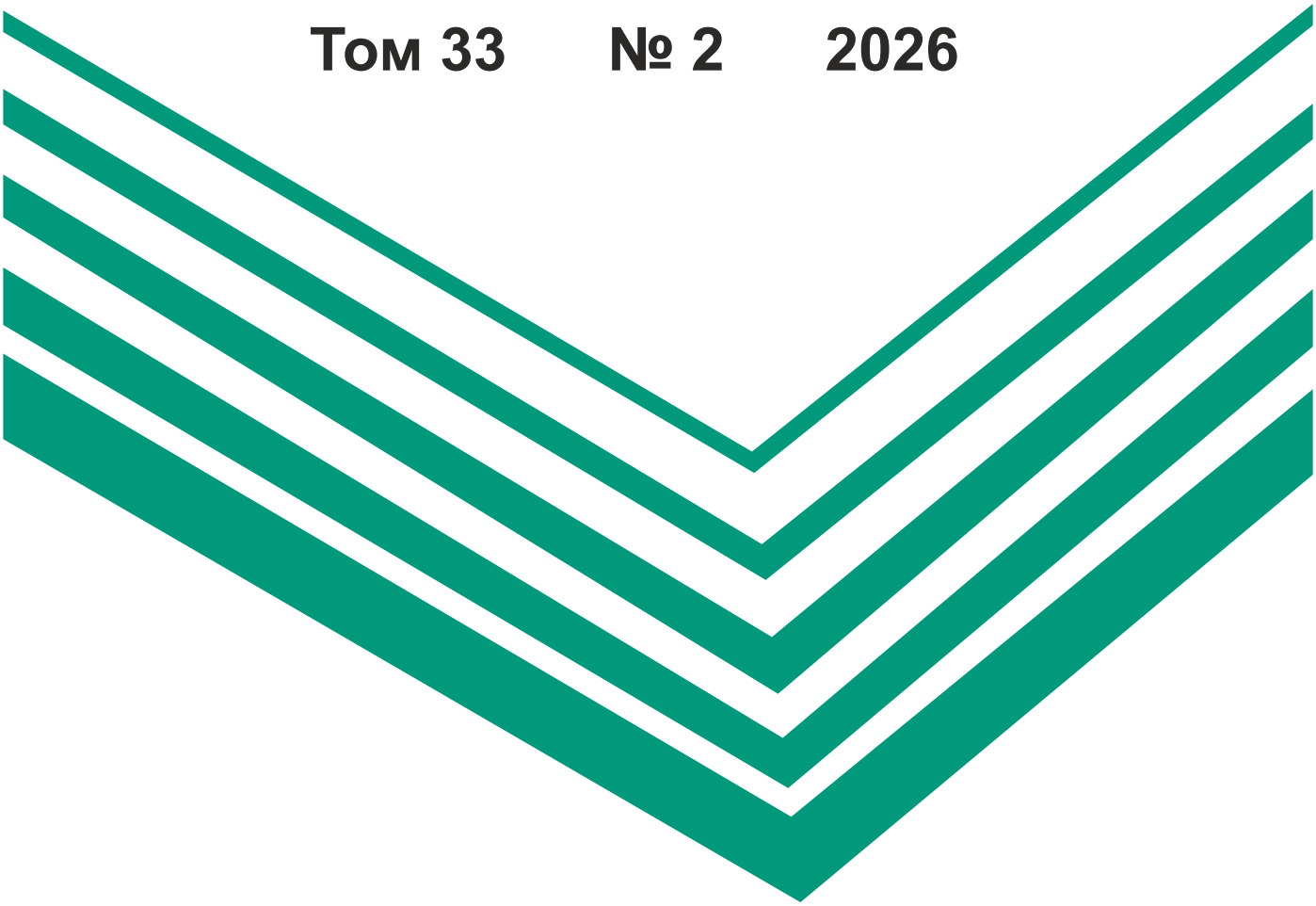
ISSN 2949-5598

# ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ

Том 33 № 2 2026



Новосибирск
Издательство Института математики

ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Апрель–июнь 2026. Т. 33, № 2. С. 97–114

УДК 519.17 DOI: 10.33048/daio.2026.33.858

# ЧЕТЫРЁХМЕРНЫЙ БУЛЕВ КУБ НЕ 3-ПРЕДСТАВИМ

*А. В. Пяткин*[1, a], *А. И. Сафарова*[2, b]

[1] Институт математики им. С. Л. Соболева,
пр. Акад. Коптюга, 4, 630090 Новосибирск, Россия
[2] Новосибирский гос. университет,
ул. Пирогова, 2, 630090 Новосибирск, Россия
E-mail: [a] artem@math.nsc.ru, [b] a.safarova1@g.nsu.ru

**Аннотация.** Граф называется 3-представимым, если существует 3-униформное слово, буквами которого являются вершины графа, причём две вершины смежны тогда и только тогда, когда соответствующие буквы чередуются в слове. В работе представлено компьютерное доказательство того, что четырёхмерный куб не 3-представим. Ил. 1, библиогр. 11.

**Ключевые слова:** представимый в виде слов граф, число представимости, четырёхмерный куб.

## Введение

Пусть задан неориентируемый граф $G = (V, E)$ без петель и кратных рёбер. Говорим, что слово $w$ над алфавитом $V$ *представляет* граф $G$, если любые две вершины $u$ и $v$ смежны тогда и только тогда, когда соответствующие буквы чередуются в слове $w$ (т. е. индуцированное буквами $u$ и $v$ подслово не содержит ни $uu$, ни $vv$). Граф называется *представимым в виде слова* (или просто *представимым*), если существует слово, которое его представляет. Это понятие введено в работе [1], и с тех пор по этой тематике опубликованы несколько десятков статей, включая обзор [2], а также книга [3].

В [1], в частности, доказано, что любой представимый граф можно представить $k$-униформным словом для некоторого $k$, т. е. таким словом, в котором каждая буква встречается ровно $k$ раз. В этом случае говорим, что граф $G$ *$k$-представим*, а минимальное такое $k$ называется *числом представимости* графа $G$ и обозначается через $\mathcal{R}(G)$. Очевидно, что класс 1-представимых графов состоит только из полных графов. Известно, что класс 2-представимых графов совпадает с классом графов пересечений хорд окружности [4]. При $k \geqslant 3$ отсутствует какая-либо

© А. В. Пяткин, А. И. Сафарова, 2026

характеризация $k$-представимых графов. Для $k = 3$ известно несколько классов 3-представимых графов: призмы [5], прямоугольные и цилиндрические решётки [6] и все представимые сплит-графы [7].

Граф называется *$k$-мерным кубом*, если его вершинами являются все $k$-мерные булевы векторы, и две вершины смежны тогда и только тогда, когда они различаются ровно в одной координате. Обозначим $k$-мерный куб через $Q_k$. В [8] доказано, что $Q_k$ $k$-представим и высказана гипотеза, что $\mathcal{R}(Q_k) = k$. При $k \leqslant 2$ это очевидно, а при $k = 3$ следует из того, что граф, получающийся при удалении одной вершины 3-мерного куба, не является графом пересечения хорд [4]. В работе [9] эта гипотеза опровергнута с помощью асимптотической оценки

$$\mathcal{R}(Q_k) = O\Big(\frac{\log k}{\log\log k}\Big)$$

при достаточно большом $k$. Однако остался открытым вопрос определения минимального $k$, для которого $k$-мерный куб является $(k-1)$-представимым графом.

Отметим, что при определении числа представимости $k$ для некоторого графа или класса графов наибольшую трудность представляет именно доказательство нижней оценки числа представимости, т. е. того факта, что он не $(k-1)$-представим. Имеется немного работ, где получены аналитические доказательства нижних оценок для числа представимости тех или иных графов. Так, в [10] доказана точная формула $\mathcal{R}(H_{n,n}) = \lceil n/2 \rceil$ при $n \geqslant 5$, где $H_{n,n}$ — граф-корона, который получается из полного двудольного графа удалением совершенного паросочетания. В [11] получены нижние оценки $\mathcal{R}(Q_k) \geqslant \Omega(\log\log k)$ для $k$-мерного куба и $\mathcal{R}(G) \geqslant n/2 - o(n)$ для почти любого двудольного графа $G$, имеющего по $n$ вершин в каждой доле.

Основным результатом статьи является

**Теорема 1.** *Граф 4-мерного куба $Q_4$ не 3-представим.*

Доказательство этой теоремы, основанное на нетривиальном компьютерном переборе, представлено в следующем разделе.

## 1. Основной результат

Сначала введём некоторые обозначения, которые будут использоваться далее. Очевидно, что любое слово $w$ представляет однозначно определённый граф, обозначаемый через $G(w)$. Пусть $x$ — буква $k$-униформного слова $w$. Тогда через $x_i$, $i = 1, \ldots, k$ обозначим $i$-ю копию буквы $x$ при чтении слова $w$ слева направо. Запись $x_i < y_j$ означает, что $i$-я копия буквы $x$ встречается в слове $w$ раньше (левее), чем $j$-я копия буквы $y$.

Тогда условие чередования букв $x$ и $y$ в слове $w$ можно записать в следующем виде:

$$(x_i < y_i < x_{i+1} \text{ для всех } i) \quad \text{или} \quad (y_i < x_i < y_{i+1} \text{ для всех } i).$$

Для произвольного подмножества букв $A \subset V$ через $w|_A$ обозначим подслово $w$, порождённое буквами из $A$. Иначе говоря, слово $w|_A$ получается из $w$ вычёркиванием всех букв из $V \setminus A$.

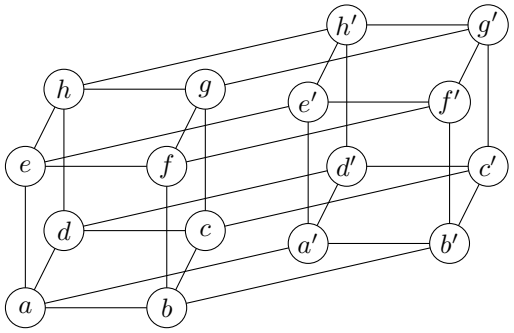



*Рис. 1.* Четырёхмерный куб $Q_4$

Обозначим так, как показано на рис. 1, вершины четырёхмерного куба $Q_4$. Для вершины $x \in V(Q_4)$ обозначим через $N(x)$ множество её соседей и положим $N[x] = x \cup N(x)$. Заметим, что если $y \in N(x)$, то между любыми $x_i$ и $x_{i+1}$ должна встречаться ровно одна копия буквы $y$. Докажем простое, но полезное

**Утверждение 1.** *Пусть $w$ — $k$-униформное слово, содержащее букву $x$, и $w_i = x_i \dots x_{i+1}$ — подслово слова $w$, заключённое между вхождениями $x_i$ и $x_{i+1}$, $i = 1, \dots, k-1$. В этом случае вершины $x$ и $y$ смежны в графе $G(w)$ тогда и только тогда, когда $y$ присутствует в каждом из слов $w_i$ ровно один раз.*

Доказательство. Если буква $y$ встречается в $w_i$ более одного раза, то подслово $w|_{\{x,y\}}$ содержит $yy$. Если буква $y$ не встречается в $w_i$, то $w|_{\{x,y\}}$ содержит $xx$. В обоих случаях $x$ и $y$ несмежны. Если каждое $w_i$ содержит по одной копии буквы $y$, то оставшаяся $k$-я копия $y$ в слове $w$ должна находиться перед $x_1$ или после $x_k$. Тогда $w|_{\{x,y\}} = (yx)^k$ или $w|_{\{x,y\}} = (xy)^k$ соответственно, а значит, вершины $x$ и $y$ смежны в $G(w)$. Утверждение 1 доказано.

Заметим, что буквы $x$ и $y$ чередуются в слове $w$ тогда и только тогда, когда они чередуются в его подслове $x_1 \dots x_k$.

**1.1. Поиск в глубину.** Наша задача — показать, что не существует 3-униформного слова $w$ над алфавитом $\{a, b, \dots, h, a', \dots, h'\}$, представляющего $Q_4$. Отметим, что всего имеется

$$\frac{48!}{6^{16}} \approx 4 \cdot 10^{48}$$

3-униформных слов длины 48, что делает полный их перебор совершенно нереальным (под полным перебором понимается алгоритм поиска в глубину, при котором на каждом шаге в слово добавляется одна из букв алфавита, которая ранее встречалась в слове менее трёх раз). Однако есть несколько ограничений и соображений, позволяющих исключить некоторые слова и варианты ветвления как очевидно не подходящие или необязательные для рассмотрения.

1. Поскольку при циклическом сдвиге (а также при развороте задом наперёд) 3-униформного слова представляемый им граф не изменяется, можно считать, что искомое слово начинается с $a$.

2. Из определения смежности следует, что если в слове появилась $x_i$, то $x_{i+1}$ не может появиться раньше, чем встретятся все буквы из $N(x)$.

3. Если в строящемся слове $w'$ появилась буква $x_3$, то по утверждению 1 можно вычислить множество $N$ всех соседей буквы $x$ в графе $G(w)$ для любого слова $w$, префиксом которого является $w'$. Если $N \neq N(x)$, то очевидно, что дальнейшее ветвление можно не рассматривать.

Выполнение п. 2 гарантирует, что каждая вершина $x$ останется смежной со всеми своими соседями, а п. 3 — что у неё не появится лишних соседей.

Поиск 3-униформного слова, представляющего граф $Q_4$, начнём следующим алгоритмом поиска в глубину с учётом ограничений 1–3. В алгоритме используются такие обозначения: word — текущее строящееся слово, $\mathrm{cnt}[x]$ — число неиспользованных в слове word копий буквы $x$, $\mathrm{forbid}[x]$ — список букв, которые должны появиться в слове word согласно ограничению 2, прежде чем можно будет добавить букву $x$, wordList — искомый список слов, представляющих граф $G$.

**Алгоритм 1. Поиск в глубину с ограничениями**

Вход: граф $G$, заданный в виде множества вершин $V$ (оно же является алфавитом искомого слова) и множеств соседей $N_G(x)$ для каждой вершины $x$, число представимости $k$.

Выход: список найденных слов wordList.

Шаг 1. Положить

$$\mathrm{word} = a, \quad \mathrm{cnt}[a] = k - 1, \quad \mathrm{forbid}[a] = N_G(a), \quad \mathrm{wordList} = \varnothing.$$

Для каждой буквы $x \neq a$ положить $\mathrm{cnt}[x] = k$, $\mathrm{forbid}[x] = \varnothing$.

Шаг 2. Запустить процедуру search(word, cnt, forbid).

**Процедура** search(word, cnt, forbid)

Шаг 1. Если $|\text{word}| = k|V|$, то добавить word в wordList и перейти на шаг 7.

Шаг 2. Для каждой буквы $x \in V$ проверить условия $\text{forbid}[x] = \varnothing$ и $\text{cnt}[x] > 0$ . Если выполняются, то перейти на шаг 3, иначе — на шаг 8.

Шаг 3. Положить

$$\text{word} = \text{word} + x, \quad \text{cnt}[x] = \text{cnt}[x] - 1, \quad \text{forbid}[x] = N_G(x).$$

Для каждой буквы $y \in N_G(x)$ положить $\text{forbid}[y] = \text{forbid}[y] - x$.

Шаг 4. Если $\text{cnt}[x] > 0$, то перейти на шаг 6, иначе — на шаг 5.

Шаг 5. Для каждой буквы $y \in V \setminus N_G[x]$ проверить, чередуются ли буквы $x$ и $y$ в текущем слове word. Если ответ «да» хотя бы для одной буквы $y$, то перейти на шаг 7, иначе — на шаг 6.

Шаг 6. Запустить процедуру search(word, cnt, forbid).

Шаг 7. Удалить последнюю букву $x$ из word. Положить $\text{cnt}[x] = \text{cnt}[x] + 1$, $\text{forbid}[x] = \varnothing$. Для всех $y \in N_G(x)$, удовлетворяющих условию $\text{cnt}[y] < k$, положить $\text{forbid}[y] = \text{forbid}[y] + x$.

Шаг 8. Перейти к следующей букве (из цикла на шаге 2).

Отметим, что выполнение ограничения 3 обеспечивается на шаге 5 рекурсивной процедуры поиска. Это гарантирует корректность алгоритма 1, поскольку для каждого слова $w$, помещённого в список на шаге 1, множество соседей каждой вершины $x$ в графе $G(w)$ совпадает с $N_G(x)$.

При попытке применить алгоритм 1 к графу $Q_4$ выяснилось, что за 10 часов работы компьютера меняется лишь 16-й символ префикса искомого слова. Поскольку число префиксов длины 15, удовлетворяющих условиям 1–3, можно грубо оценить снизу как $15! \approx 10^{12}$, очевидно, что алгоритм 1 не сможет доказать теорему 1 за приемлемое время. Тем не менее, с помощью этого алгоритма удалось получить списки всех слов, начинающихся с буквы $a$, которые представляют графы $Q_2$ и $Q_3$. Для $Q_2$ список состоит из 36 слов длины 12, а для $Q_3$ — из 82 944 слов длины 24. Эти списки сыграли важную роль в получении компьютерного доказательства теоремы 1, приведённого в п. 1.3.

**1.2. Эквивалентность и скрещивание слов.** Пусть заданы слова $w_1$ и $w_2$ над непересекающимися алфавитами $A$ и $B$ соответственно. Говорим, что слово $w$ над алфавитом $A \cup B$ получено *скрещиванием* слов $w_1$ и $w_2$, если $w|_A = w_1$ и $w|_B = w_2$. Смысл этого понятия заключается в следующем. Пусть множество вершин графа $G(w)$, представленного словом $w$, разбито на непересекающиеся подмножества $A$ и $B$.

Тогда очевидно, что индуцированные подграфы $G[A]$ и $G[B]$ представлены словами $w_1 = w|_A$ и $w_2 = w|_B$ соответственно, а само слово $w$ получено скрещиванием слов $w_1$ и $w_2$. Это порождает следующую идею доказательства теоремы 1: попробовать угадать слова, представляющие некоторые подграфы $A$ и $B$, а затем построить все их возможные скрещивания. Однако, для реализации этой идеи следует существенно сократить число слов, рассматриваемых для скрещивания.

*Разворотом* слова $w = x_1x_2\dots x_n$ называется слово $x_1x_nx_{n-1}\dots x_2$, получающееся из $w$ с помощью инверсии и циклического сдвига на одну позицию вправо. Назовём два $k$-униформных слова *тривиально эквивалентными*, если одно из них переводится в другое с помощью операций разворота и/или циклического сдвига. Очевидно, что два тривиально эквивалентных слова представляют один и тот же граф, а среди таких слов, начинающихся с буквы $a$, каждый класс эквивалентности состоит из $2k$ слов. Биекция $F\colon V \to V$ называется *автоморфизмом* графа $G$, если она сохраняет смежность и несмежность вершин, т. е. $xy \in E$ тогда и только тогда, когда $F(x)F(y) \in E$. Пусть два слова $w_1 = x_1x_2\dots x_n$ и $w_2$ над алфавитом $V$ начинаются с буквы $a$ и представляют граф $G = (V, E)$. Говорим, что они *эквивалентны*, если существует такой автоморфизм $F$ графа $G$, что слово $F(x_1)F(x_2)\dots F(x_n)$ тривиально эквивалентно слову $w_2$. Поскольку автоморфизм — это просто переименование вершин графа, два эквивалентных слова представляют один и тот же граф. Для иллюстрации вышеприведённой концепции докажем аналитически

**Утверждение 2.** *Существует* 36 3-*униформных слов, начинающихся с буквы* $a$ *и представляющих граф* $Q_2$ (4-*цикл*)*, среди которых ровно два неэквивалентных.*

Доказательство. Сначала докажем вторую часть утверждения 2. Пусть 3-униформное слово $w = aI_1aI_2aI_3$ представляет 4-цикл $abcd$. Тогда каждая из букв $b$ и $d$ встречается в каждом из подслов $I_j$ ровно один раз, а буква $c$ встречается в каком-то из $I_j$ более одного раза. Ясно, что никакое из $I_j$ не может содержать три копии $c$, так как тогда она не будет смежна с $b$ и $d$. С помощью операций циклического сдвига и/или разворота можно добиться того, что интервалы $I_1, I_2$ и $I_3$ содержат 0, 1 и 2 копии буквы $c$ соответственно. Можно считать, что в $I_1$ буквы $b$ и $d$ идут в порядке $bd$ (иначе применим автоморфизм, меняющий в цикле вершины $b$ и $d$ местами). Тогда $w = abdac\{b,d\}ac\{b,d\}c$. Поскольку $b$ и $d$ несмежны, хотя бы в одном из множеств $\{b,d\}$ буквы должны идти в порядке $db$. В итоге имеем три возможных неэквивалентных слова:

$$w_1 = abdacbdacdbc, \quad w_2 = abdacdbacbdc, \quad w_3 = abdacdbacdbc.$$

Заметим, однако, что слово $w_3$ эквивалентно $w_1$: применим автоморфизм, меняющий в цикле вершины $a$ и $c$ местами, и прочитаем полученное слово задом наперёд. Для доказательства неэквивалентности слов $w_1$ и $w_2$ воспользуемся следующим приёмом. Назовём *следом* буквы $x$ в слове $w$ интервал между двумя её копиями, который содержит только двух её соседей. Нетрудно видеть, что каждая из букв имеет ровно один след в каждом из слов $w_1$ и $w_2$ (следы обозначены скобками):

$$w_1 = (abd)acbd(ac)(dbc), \quad w_2 = a(bdac)db(acbd)c.$$

При этом в слове $w_1$ все 4 следа идут подряд, тогда как в слове $w_2$ они разбиты на две отдельные группы по 2 следа. Поскольку операции сдвига, разворота и автоморфизма никак не влияют на взаимное расположение следов, слова $w_1$ и $w_2$ неэквивалентны.

Для доказательства первой части утверждения 2 найдём мощности классов эквивалентности слов $w_1$ и $w_2$. Заметим, что существует ровно 8 автоморфизмов графа $Q_2$ (4 способа выбрать букву $a$ и 2 способа выбрать её соседей). Каждый из автоморфизмов порождает 6 тривиально эквивалентных слов, начинающихся с буквы $a$ и представляющих $Q_2$, что даёт верхнюю оценку 48 на размер класса эквивалентности, но при этом некоторые автоморфизмы могут перевести слово в тривиально эквивалентное ему самому. Ясно, что каждый такой автоморфизм сокращает размер класса эквивалентности вдвое. Для слова $w_1$ таким автоморфизмом является $F(a) = d$, $F(b) = c$, $F(c) = b$, $F(d) = a$ (при прочтении полученного слова задом наперёд начиная с 10-й буквы получится $w_1$). У слова $w_2$ имеется два таких автоморфизма. Первый из них тот же, что и для $w_1$ (только читать слово задом наперёд надо с 6-й буквы), а второй $F(a) = c$, $F(b) = d$, $F(c) = a$, $F(d) = b$ (слово читается задом наперёд). Непосредственной проверкой можно убедиться, что остальные автоморфизмы не переводят эти слова в тривиально эквивалентные себе. Таким образом, имеется 24 слова, эквивалентных $w_1$, и 12 слов, эквивалентных $w_2$, всего 36 3-униформных слов, начинающихся с буквы $a$ и представляющих $Q_2$. Утверждение 2 доказано.

Для получения аналогичного утверждения относительно графа $Q_3$ пришлось применить компьютерный метод и действовать в обратном порядке: сначала с помощью алгоритма 1 получен список fullList из 82 944 3-униформных слов, начинающихся с буквы $a$ и представляющих $Q_3$, а потом к нему применён следующий алгоритм, последовательно удаляющий из списка эквивалентные слова и выдающий на выходе список неэквивалентных слов unList. Для автоморфизма $F$ графа $G = (V, E)$ и слова word над алфавитом $V$ через $F(\text{word})$ обозначается слово, полученное заменой каждой буквы $x$ слова word на $F(x)$.

**Алгоритм 2. Получение списка неэквивалентных слов**

Вход: список fullList $k$-униформных слов, начинающихся с буквы $a$ и представляющих граф $G$. Список автоморфизмов $\operatorname{Aut} G$ графа $G$.

Выход: список попарно неэквивалентных слов unList.

Шаг 1. Положить unList $= \varnothing$.

Шаг 2. Выбрать текущее слово word из списка fullList и добавить его в unList.

Шаг 3. Для каждого автоморфизма $F \in \operatorname{Aut} G$ найти все $2k$ слов, тривиально эквивалентных $F(\text{word})$, и удалить их из списка fullList.

Шаг 4. Если fullList $\neq \varnothing$, то перейти на шаг 2.

Для построения списка автоморфизмов $\operatorname{Aut} G$ куба $Q_3$ используется следующая процедура. Сначала выбирается образ $x = F(a)$ буквы $a$ (8 вариантов). Затем рассматриваются все возможные варианты назначения образов букв $b, d$ и $e$ среди соседей $N(x)$ буквы $x$ в кубе $Q_3$ (6 вариантов). Тогда образы букв $c, f$ и $h$ определяются однозначно как попарные общие соседи букв $F(b), F(d)$ и $F(e)$, а последняя буква будет образом $g$. Всего получается 48 автоморфизмов.

Применяя алгоритм 1 к кубу $Q_3$ и подавая найденный им список слов в качестве входа алгоритма 2, получаем машинное доказательство следующего утверждения.

**Утверждение 3.** *Существует* 82 944 3*-униформных слов, начинающихся с буквы* $a$ *и представляющих граф* $Q_3$*, среди которых ровно* 308 *попарно неэквивалентных.*

Список попарно неэквивалентных слов из утверждения 3 приводится в прил. А.

Далее выпишем алгоритм скрещивания двух слов, играющий ключевую роль в доказательстве теоремы 1. Он работает очень схоже с алгоритмом 1, но на каждом шаге выбор добавляемой буквы ограничен лишь двумя вариантами, а именно, текущими буквами в скрещиваемых словах (их позиции указаны переменными $p_1$ и $p_2$). Остальные переменные (cnt, forbid, word и wordList) определены так же, как и в алгоритме 1.

**Алгоритм 3. Скрещивание двух слов**

Вход: два $k$-униформных слова $w_1$ и $w_2$ над алфавитами $A$ и $B$, $A \cap B = \varnothing$, причём $w_1$ начинается с буквы $a \in A$. Граф $G$ с множеством вершин $V = A \cup B$, чьи подграфы $G[A]$ и $G[B]$ представлены словами $w_1$ и $w_2$ соответственно.

Выход: список найденных скрещиваний wordList.

Шаг 1. Положить

$$\text{word} = a, \quad \text{cnt}[a] = k-1, \quad \text{forbid}[a] = N_G(a), \quad \text{wordList} = \varnothing.$$

Для каждой буквы $x \neq a$ положить $\text{cnt}[x] = k$, $\text{forbid}[x] = \varnothing$. Установить счётчики $p_1 = 2$ и $p_2 = 1$.

Шаг 2. Запустить процедуру mix(word, cnt, forbid, $p_1$, $p_2$).

**Процедура** mix(word, cnt, forbid, $p_1$, $p_2$)

Шаг 1. Если $|\text{word}| = k|V|$, то добавить word в wordList и перейти на шаг 8.

Шаг 2. Задать множество $\text{next} = \{w_1(p_1), w_2(p_2)\}$.

Шаг 3. Для каждой буквы $x \in \text{next}$ проверить условия $\text{forbid}[x] = \varnothing$ и $\text{cnt}[x] > 0$ . Если выполняются, то перейти на шаг 4, иначе — на шаг 9.

Шаг 4. Положить

$$\text{word} = \text{word} + x, \quad \text{cnt}[x] = \text{cnt}[x] - 1, \quad \text{forbid}[x] = N_G(x), \quad p_i = p_i + 1,$$

где $i = 1$, если $x \in A$, и $i = 2$, если $x \in B$. Для каждой буквы $y \in N_G(x)$ положить $\text{forbid}[y] = \text{forbid}[y] - x$.

Шаг 5. Если $\text{cnt}[x] > 0$, то перейти на шаг 7, иначе — на шаг 6.

Шаг 6. Для каждой буквы $y \in V \setminus N_G[x]$ проверить, чередуются ли буквы $x$ и $y$ в текущем слове word. Если ответ «да» хотя бы для одной буквы $y$, то перейти на шаг 8, иначе — на шаг 7.

Шаг 7. Запустить процедуру mix(word, cnt, forbid, $p_1$, $p_2$).

Шаг 8. Удалить последнюю букву $x$ из word. Положить $\text{cnt}[x] = \text{cnt}[x] + 1$, $\text{forbid}[x] = \varnothing$, $p_i = p_i - 1$. Для всех $y \in N_G(x)$, удовлетворяющих условию $\text{cnt}[y] < k$, положить $\text{forbid}[y] = \text{forbid}[y] + x$.

Шаг 9. Перейти к следующей букве (из цикла на шаге 3).

Отметим, что алгоритм 3 работает гораздо быстрее алгоритма 1, поскольку на каждом шаге ветвления имеется не более двух вариантов, тогда как в алгоритме 1 их число может доходить до 15. Также оказалось, что скрещивание длинного слова с коротким происходит значительно быстрее, чем двух слов средней длины. Это предопределило технику поиска гипотетического слова $w$, представляющего $Q_4$, описанную в следующем пункте.

**1.3. Компьютерное доказательство теоремы 1.** С помощью алгоритма 3 выполнены последовательно следующие шаги.

1. Построены все возможные скрещивания 308 неэквивалентных слов из утверждения 3 (см. прил. А) со всеми 144 словами, представляющими цикл $a'b'c'd'$ (см. прил. Б). В результате получился список $\text{list}_1$ из 12 590 580 слов на букву $a$, представляющих $Q_4 \setminus \{e', f', g', h'\}$.

2. Построены все возможные скрещивания слов из списка $\mathrm{list}_1$ со списком из двух возможных слов $\{e'f'e'f'e'f',\ f'e'f'e'f'e'\}$, представляющих ребро $e'f'$. В результате получился список $\mathrm{list}_2$ из 142 597 981 слов на букву $a$, представляющих $Q_4 \setminus \{g', h'\}$.

3. Построены все возможные скрещивания слов из списка $\mathrm{list}_2$ с единственным словом $h'h'h'$, представляющих одновершинный подграф $h'$. В результате получился список $\mathrm{list}_3$ из 9 673 917 слов на букву $a$, представляющих $Q_4 \setminus \{g'\}$.

4. Наконец, построены все возможные скрещивания слов из списка $\mathrm{list}_3$ с единственным словом $g'g'g'$, представляющим одновершинный подграф $g'$. Итоговый список оказался пустым.

Отсюда следует, что граф $Q_4$ не 3-представим. Действительно, предположим, что найдётся 3-униформное слово $w$, начинающееся с буквы $a$, которое его представляет. Пусть $A = \{a, b, \ldots, h\}$. Тогда подслово $w_1 = w|_A$ представляет подграф $Q_3$. По утверждению 3 найдётся такой автоморфизм $F$ графа $Q_3$, что $F(w_1)$ тривиально эквивалентен одному из 308 слов, приведённых в прил. А. Заметим, что автоморфизм $F$ подграфа $Q_3$ однозначно расширяется до автоморфизма $F'$ графа $Q_4$, поскольку каждая вершина из $A$ имеет ровно одного соседа в $\{a', b', \ldots, h'\}$, и наоборот. Тогда слово $F'(w)$ можно с помощью операций циклического сдвига и/или разворота превратить в такое слово $w'$, начинающееся на $a$, что подслово $w'|_A$ содержится в списке из прил. А. Пусть $B = \{a', b', c', d'\}$. Тогда подслово $w'|_B$, представляющее 4-цикл $a'b'c'd'$, должно присутствовать в списке из прил. Б. Значит, подслово $w'|_{A\cup B}$ лежит в списке $\mathrm{list}_1$. Аналогично подслово $w'|_{V\setminus\{g',h'\}}$ должно лежать в списке $\mathrm{list}_2$, а подслово $w'_1 = w'|_{V\setminus\{g'\}}$ — в списке $\mathrm{list}_3$, но тогда слово $w'$ должно получаться скрещиванием $w'_1$ со словом $g'g'g'$, а таких слов нет. Теорема 1 доказана.

Компьютерное доказательство теоремы 1 заняло порядка 300 часов машинного времени. Авторы располагают списками $\mathrm{list}_1, \mathrm{list}_2, \mathrm{list}_3$, общий объём которых составляет свыше 7,2 ГБ.

## Заключение

В работе исследована задача определения числа представимости четырёхмерного куба. С помощью нетривиального компьютерного поиска показано, что данное число равно 4. Проблема поиска полностью аналитического доказательства этого факта остаётся открытой. Также открыта более общая проблема, сформулированная в [9]: определить минимальное число $k$, для которого $k$-мерный куб будет $(k-1)$-представимым графом.

**Финансирование работы**

Исследование выполнено в рамках государственного задания Института математики им. С. Л. Соболева (проект № FWNF–2026–0021). Дополнительных грантов на проведение или руководство этим исследованием получено не было.

**Конфликт интересов**

Авторы заявляют, что у них нет конфликта интересов.

**Литература**

1. **Kitaev S. V., Pyatkin A. V.** On representable graphs // J. Autom. Lang. Comb. 2008. V. 13, No. 1. P. 45–54.
2. **Китаев С. В., Пяткин А. В.** Графы, представимые в виде слов. Обзор результатов // Дискрет. анализ и исслед. операций. 2018. Т. 25, № 2. С. 19–53.
3. **Kitaev S. V., Lozin V. V.** Words and graphs. Cham: Springer, 2015. 264 p.
4. **Bouchet A.** Circle graph obstructions // J. Comb. Theory, Ser. B. 1994. V. 60, No. 1. P. 107–144.
5. **Kitaev S. V.** On graphs with representation number 3 // J. Autom. Lang. Comb. 2013. V. 18, No. 2. P. 97–112.
6. **Alshammari N. S., Kitaev S. V., Pyatkin A. V.** On the representation number of grid graphs and cylindric grid graphs. Ithaca, NY, 2025. 10 p. (e-Print Archive / Cornell Univ.; arXiv:2507.16469). DOI: `10.48550/arXiv.2507.16469`.
7. **Dwary T., Mozhui K., Krishna K. V.** Representation number of word-representable split graphs. Ithaca, NY, 2025. 14 p. (e-Print Archive / Cornell Univ.; arXiv:2502.00872). DOI: `10.48550/arXiv.2502.00872`.
8. **Broere B., Zantema H.** The $k$-dimensional cube is $k$-representable // J. Autom. Lang. Comb. 2019. V. 24, No. 1. P. 3–12.
9. **Hefty Z., Horn P., Muir C., Owens A.** Word-representable graphs: Orientations, posets, and bounds // Electron. J. Comb. 2024. V. 31, No. 4. Article ID P4.2. 26 p.
10. **Glen M., Kitaev S. V., Pyatkin A. V.** On the representation number of a crown graph // Discrete Appl. Math. 2018. V. 244. P. 89–93.
11. **Hefty Z., Horn P., Muir C., Owens A.** Word-representation numbers of graphs: Bottlenecks and bounds // J. Comb. Theory, Ser. A. 2026. V. 223. Article ID 106215. 25 p.

## Приложение A. Попарно неэквивалентные 3-униформные слова, начинающиеся на $a$ и представляющие $Q_3$

1) $abdceafbhedagchdfebgfhcg$
2) $abdceafbhedagchdfebghfcg$
3) $abdceafbhedagchdfegbfhcg$
4) $abdceafbhedagchdfegbhfcg$
5) $abdceafbhedagchdfeghbfcg$
6) $abdceafbhedagchdfgebfhcg$

7) $abdceafbhedagchdfgebhfcg$
8) $abdceafbhedagchdfgehbfcg$
9) $abdceafbhedaghcdfebgfchg$
10) $abdceafbhedaghcdfebgfhcg$
11) $abdceafbhedaghcdfebghfcg$
12) $abdceafbhedaghcdfegbfchg$
13) $abdceafbhedaghcdfegbfhcg$
14) $abdceafbhedaghcdfegbhfcg$
15) $abdceafbhedaghcdfeghbfcg$
16) $abdceafbhedaghcdfgebfchg$
17) $abdceafbhedaghcdfgebfhcg$
18) $abdceafbhedaghcdfgebhfcg$
19) $abdceafbhedaghcdfgehbfcg$
20) $abdceafbhedgachdfebgfhcg$
21) $abdceafbhedgachdfebghfcg$
22) $abdceafbhedgachdfegbfhcg$
23) $abdceafbhedgachdfegbhfcg$
24) $abdceafbhedgachdfeghbfcg$
25) $abdceafbhedgachdfgebfhcg$
26) $abdceafbhedgachdfgebhfcg$
27) $abdceafbhedgachdfgehbfcg$
28) $abdceafbhedgahcdfebgfchg$
29) $abdceafbhedgahcdfebgfhcg$
30) $abdceafbhedgahcdfebghfcg$
31) $abdceafbhedgahcdfegbfchg$
32) $abdceafbhedgahcdfegbfhcg$
33) $abdceafbhedgahcdfegbhfcg$
34) $abdceafbhedgahcdfeghbfcg$
35) $abdceafbhedgahcdfgebfchg$
36) $abdceafbhedgahcdfgebfhcg$
37) $abdceafbhedgahcdfgebhfcg$
38) $abdceafbhedgahcdfgehbfcg$
39) $abdceafbhedghacdfebgfchg$
40) $abdceafbhedghacdfebgfhcg$
41) $abdceafbhedghacdfebghfcg$
42) $abdceafbhedghacdfegbfchg$
43) $abdceafbhedghacdfegbfhcg$
44) $abdceafbhedghacdfegbhfcg$
45) $abdceafbhedghacdfeghbfcg$
46) $abdceafbhedghacdfgebfchg$
47) $abdceafbhedghacdfgebfhcg$
48) $abdceafbhedghacdfgebhfcg$
49) $abdceafbhedghacdfgehbfcg$
50) $abdceafbhegdachdfebgfhcg$
51) $abdceafbhegdachdfebghfcg$
52) $abdceafbhegdachdfegbfhcg$
53) $abdceafbhegdachdfegbhfcg$
54) $abdceafbhegdachdfeghbfcg$
55) $abdceafbhegdachdfgebfhcg$
56) $abdceafbhegdachdfgebhfcg$
57) $abdceafbhegdachdfgehbfcg$
58) $abdceafbhegdahcdfebgfchg$
59) $abdceafbhegdahcdfebgfhcg$
60) $abdceafbhegdahcdfebghfcg$
61) $abdceafbhegdahcdfegbfchg$
62) $abdceafbhegdahcdfegbfhcg$
63) $abdceafbhegdahcdfegbhfcg$
64) $abdceafbhegdahcdfeghbfcg$
65) $abdceafbhegdahcdfgebfchg$
66) $abdceafbhegdahcdfgebfhcg$
67) $abdceafbhegdahcdfgebhfcg$
68) $abdceafbhegdahcdfgehbfcg$
69) $abdceafbhegdhacdfebgfchg$
70) $abdceafbhegdhacdfebgfhcg$
71) $abdceafbhegdhacdfebghfcg$
72) $abdceafbhegdhacdfegbfchg$
73) $abdceafbhegdhacdfegbfhcg$
74) $abdceafbhegdhacdfegbhfcg$
75) $abdceafbhegdhacdfeghbfcg$
76) $abdceafbhegdhacdfgebfchg$
77) $abdceafbhegdhacdfgebfhcg$
78) $abdceafbhegdhacdfgebhfcg$
79) $abdceafbhegdhacdfgehbfcg$
80) $abdceafbhegfdachdebghcfg$
81) $abdceafbhegfdachdebghfcg$
82) $abdceafbhegfdachdegbhcfg$
83) $abdceafbhegfdachdegbhfcg$
84) $abdceafbhegfdachdeghbfcg$
85) $abdceafbhegfdachdgebhcfg$
86) $abdceafbhegfdachdgebhfcg$
87) $abdceafbhegfdachdgehbfcg$
88) $abdceafbhegfdachedbghcfg$
89) $abdceafbhegfdachedbghfcg$
90) $abdceafbhegfdachedgbhcfg$

91) $abdceafbhegfdachedgbhfcg$
92) $abdceafbhegfdachedghbfcg$
93) $abdceafbhegfdahcdebgchfg$
94) $abdceafbhegfdahcdebghcfg$
95) $abdceafbhegfdahcdebghfcg$
96) $abdceafbhegfdahcdegbchfg$
97) $abdceafbhegfdahcdegbhcfg$
98) $abdceafbhegfdahcdegbhfcg$
99) $abdceafbhegfdahcdeghbfcg$
100) $abdceafbhegfdahcdgebchfg$
101) $abdceafbhegfdahcdgebhcfg$
102) $abdceafbhegfdahcdgebhfcg$
103) $abdceafbhegfdahcdgehbfcg$
104) $abdceafbhegfdahcedbgchfg$
105) $abdceafbhegfdahcedbghcfg$
106) $abdceafbhegfdahcedbghfcg$
107) $abdceafbhegfdahcedgbchfg$
108) $abdceafbhegfdahcedgbhcfg$
109) $abdceafbhegfdahcedgbhfcg$
110) $abdceafbhegfdahcedghbfcg$
111) $abdceafbhegfdahecdbgchfg$
112) $abdceafbhegfdahecdbghcfg$
113) $abdceafbhegfdahecdbghfcg$
114) $abdceafbhegfdahecdgbchfg$
115) $abdceafbhegfdahecdgbhcfg$
116) $abdceafbhegfdahecdgbhfcg$
117) $abdceafbhegfdahecdghbfcg$
118) $abdceafbhegfdhacdebgchfg$
119) $abdceafbhegfdhacdebghcfg$
120) $abdceafbhegfdhacdebghfcg$
121) $abdceafbhegfdhacdegbchfg$
122) $abdceafbhegfdhacdegbhcfg$
123) $abdceafbhegfdhacdegbhfcg$
124) $abdceafbhegfdhacdeghbfcg$
125) $abdceafbhegfdhacdgebchfg$
126) $abdceafbhegfdhacdgebhcfg$
127) $abdceafbhegfdhacdgebhfcg$
128) $abdceafbhegfdhacdgehbfcg$
129) $abdceafbhegfdhacedbgchfg$
130) $abdceafbhegfdhacedbghcfg$
131) $abdceafbhegfdhacedbghfcg$
132) $abdceafbhegfdhacedgbchfg$
133) $abdceafbhegfdhacedgbhcfg$
134) $abdceafbhegfdhacedgbhfcg$
135) $abdceafbhegfdhacedghbfcg$
136) $abdceafbhgedachdfebgfhcg$
137) $abdceafbhgedachdfebghfcg$
138) $abdceafbhgedachdfegbfhcg$
139) $abdceafbhgedachdfegbhfcg$
140) $abdceafbhgedachdfeghbfcg$
141) $abdceafbhgedahcdfebgfchg$
142) $abdceafbhgedahcdfebgfhcg$
143) $abdceafbhgedahcdfebghfcg$
144) $abdceafbhgedahcdfegbfchg$
145) $abdceafbhgedahcdfegbfhcg$
146) $abdceafbhgedahcdfegbhfcg$
147) $abdceafbhgedahcdfeghbfcg$
148) $abdceafbhgedhacdfebgfchg$
149) $abdceafbhgedhacdfebgfhcg$
150) $abdceafbhgedhacdfebghfcg$
151) $abdceafbhgedhacdfegbfchg$
152) $abdceafbhgedhacdfegbfhcg$
153) $abdceafbhgedhacdfegbhfcg$
154) $abdceafbhgedhacdfeghbfcg$
155) $abdceafbhgefdachdebghcfg$
156) $abdceafbhgefdachdebghfcg$
157) $abdceafbhgefdachdegbhcfg$
158) $abdceafbhgefdachdegbhfcg$
159) $abdceafbhgefdachdeghbfcg$
160) $abdceafbhgefdachedbghcfg$
161) $abdceafbhgefdachedbghfcg$
162) $abdceafbhgefdachedgbhcfg$
163) $abdceafbhgefdachedgbhfcg$
164) $abdceafbhgefdachedghbfcg$
165) $abdceafbhgefdahcdebghcfg$
166) $abdceafbhgefdahcdebghfcg$
167) $abdceafbhgefdahcdegbhcfg$
168) $abdceafbhgefdahcdegbhfcg$
169) $abdceafbhgefdahcdeghbfcg$
170) $abdceafbhgefdahcedbghfcg$
171) $abdceafbhgefdahcedgbhcfg$
172) $abdceafbhgefdahcedgbhfcg$
173) $abdceafbhgefdahcedghbfcg$
174) $abdceafbhgefdhacdebghfcg$

175) $abdceafbhgefdhacdegbhfcg$
176) $abdceafbhgefdhacdeghbfcg$
177) $abdceafbhgefdhacedgbhfcg$
178) $abdceafbhgefdhacedghbfcg$
179) $abdceafbhgefdhaecdghbfcg$
180) $abdceafhbedaghcdfeghbfcg$
181) $abdceafhbedgachdfeghbfcg$
182) $abdceafhbedgahcdfeghbfcg$
183) $abdceafhbedghacdfeghbfcg$
184) $abdceafhbegdachdfeghbfcg$
185) $abdceafhbegdahcdfeghbfcg$
186) $abdceafhbegdhacdfeghbfcg$
187) $abdceafhbegfdachdeghbfcg$
188) $abdceafhbegfdachdgehbfcg$
189) $abdceafhbegfdachedghbfcg$
190) $abdceafhbegfdahcdeghbfcg$
191) $abdceafhbegfdahcdgehbfcg$
192) $abdceafhbegfdahcedghbfcg$
193) $abdceafhbegfdhacdeghbfcg$
194) $abdceafhbegfdhacdgehbfcg$
195) $abdceafhbegfdhacedghbfcg$
196) $abdceafhbgedachdfeghbfcg$
197) $abdceafhbgedahcdfeghbfcg$
198) $abdceafhbgedhacdfeghbfcg$
199) $abdceafhbgefdachdeghbfcg$
200) $abdceafhbgefdachedghbfcg$
201) $abdceafhbgefdahcdeghbfcg$
202) $abdceafhbgefdhacdeghbfcg$
203) $abdceafhbgefdhacedghbfcg$
204) $abdceafhdeghbacfbegfdchg$
205) $abdceafhdeghbacfbgefdchg$
206) $abdceafhdeghbacfebgfdchg$
207) $abdceafhdeghbafcbegfdchg$
208) $abdceafhdeghbafcbgefdchg$
209) $abdceafhdeghbafcebgfdchg$
210) $abdceafhdeghbafecbgfdchg$
211) $abdceafhdeghbfacbegfdchg$
212) $abdceafhdeghbfacbgefdchg$
213) $abdceafhdeghbfacebgfdchg$
214) $abdceafhdgehbacfbegfdchg$
215) $abdceafhdgehbacfebgfdchg$
216) $abdceafhdgehbafcebgfdchg$
217) $abdceafhdgehbafecbgfdchg$
218) $abdceafhdgehbfacbegfdchg$
219) $abdceafhdgehbfacebgfdchg$
220) $abdceafhebdgahcdfeghbfcg$
221) $abdceafhebdghacdfeghbfcg$
222) $abdceafhebgdachdfeghbfcg$
223) $abdceafhebgdahcdfeghbfcg$
224) $abdceafhebgdhacdfeghbfcg$
225) $abdceafhebgfdachdeghbfcg$
226) $abdceafhebgfdachdgehbfcg$
227) $abdceafhebgfdachedghbfcg$
228) $abdceafhebgfdahcdeghbfcg$
229) $abdceafhebgfdahcdgehbfcg$
230) $abdceafhebgfdhacdeghbfcg$
231) $abdceafhebgfdhacdgehbfcg$
232) $abdceafhedbaghcdfeghbfcg$
233) $abdceafhedbgahcdfeghbfcg$
234) $abdceafhedbghacdfeghbfcg$
235) $abdceafhedghbacdfbegfchg$
236) $abdceafhedghbacdfbegfhcg$
237) $abdceafhedghbacdfbeghfcg$
238) $abdceafhedghbacdfbgefchg$
239) $abdceafhedghbacdfbgefhcg$
240) $abdceafhedghbacdfbgehfcg$
241) $abdceafhedghbacdfebgfchg$
242) $abdceafhedghbacdfebgfhcg$
243) $abdceafhedghbacdfebghfcg$
244) $abdceafhedghbacfbegfdchg$
245) $abdceafhedghbacfbgefdchg$
246) $abdceafhedghbacfebgfdchg$
247) $abdceafhedghbafcbgefdchg$
248) $abdceafhedghbafcebgfdchg$
249) $abdceafhedghbafecbgfdchg$
250) $abdceafhedghbfacbgefdchg$
251) $abdceafhedghbfacebgfdchg$
252) $abdceafhegdhbacdfbegfhcg$
253) $abdceafhegdhbacdfbeghfcg$
254) $abdceafhegdhbacdfbgefchg$
255) $abdceafhegdhbacdfbgehfcg$
256) $abdceafhegdhbacdfebgfchg$
257) $abdceafhgedhbacdfbeghfcg$
258) $abdceafhgedhbacdfebgfchg$

259) $abdceafhgedhbacdfebghfcg$
260) $abdceahdfbeagcfhegdhbfcg$
261) $abdceahdfbeagcfhgedhbfcg$
262) $abdceahdfbeagfhcegdhbfcg$
263) $abdceahdfbeagfhcgedhbfcg$
264) $abdceahdfbeagfhecgdhbfcg$
265) $abdceahdfbegacfhegdhbfcg$
266) $abdceahdfbegacfhgedhbfcg$
267) $abdceahdfbegafchegdhbfcg$
268) $abdceahdfbegafchgedhbfcg$
269) $abdceahdfbegafhcegdhbfcg$
270) $abdceahdfbegafhcgedhbfcg$
271) $abdceahdfbegfachegdhbfcg$
272) $abdceahdfbegfachgedhbfcg$
273) $abdceahdfbegfahcgedhbfcg$
274) $abdceahdfbegfhacgedhbfcg$
275) $abdceahdfegbacfbhedghfcg$
276) $abdceahdfegbacfbhegdhfcg$
277) $abdceahdfegbacfbhgedhfcg$
278) $abdceahdfegbafcbhedghcfg$
279) $abdceahdfegbafcbhedghfcg$
280) $abdceahdfegbafcbhegdhcfg$
281) $abdceahdfegbafcbhegdhfcg$
282) $abdceahdfegbafcbhgedhcfg$
283) $abdceahdfegbfacbhedghcfg$
284) $abdceahdfegbfacbhedghfcg$
285) $abdceahdfegbfacbhegdhcfg$
286) $abdceahdfegbfacbhgedhcfg$
287) $abdceahdfgebafcbhedghcfg$
288) $abdceahdfgebafcbhegdhcfg$
289) $abdceahdfgebfacbhedghcfg$
290) $abdceahfdeghbacfbgefdchg$
291) $abdceahfdeghbacfebgfdchg$
292) $abdceahfdeghbafcbgefdchg$
293) $abdceahfdeghbafcebgfdchg$
294) $abdceahfdeghbfacbgefdchg$
295) $abdceahfdeghbfacebgfdchg$
296) $abdceahfdgehbacfebgfdchg$
297) $abdceahfdgehbfacebgfdchg$
298) $abdceahfedghbacdfbgefchg$
299) $abdceahfedghbacdfebgfchg$
300) $abdceahfedghbacfbgefdchg$
301) $abdceahfedghbacfebgfdchg$
302) $abdceahfedghbafcbgefdchg$
303) $abdceahfedghbfacbgefdchg$
304) $abdceahfgedhbacdfebgfchg$
305) $abdcefabhgedahcdfebgfchg$
306) $abdcefabhgedahcdfegbfchg$
307) $abdcefabhgedhacdfebgfchg$
308) $abdcefahedghbacdfebgfchg$

## Приложение Б. Все 3-униформные слова, представляющие 4-цикл

1) $abdacbdacdbc$
2) $abdacbdcadbc$
3) $abdacdbacbdc$
4) $abdacdbacdbc$
5) $abdacdbcabdc$
6) $abdacdbcadbc$
7) $abdcabdacdbc$
8) $abdcadbacbdc$
9) $abdcadbacdbc$
10) $adbacbdacbdc$
11) $adbacbdacdbc$
12) $adbacbdcabdc$
13) $adbacbdcadbc$
14) $adbacdbacbdc$
15) $adbacdbcabdc$
16) $adbcabdacbdc$
17) $adbcabdacdbc$
18) $adbcadbacbdc$
19) $acbdacbdcadb$
20) $acbdacdbcabd$
21) $acbdacdbcadb$
22) $acbdcabdacdb$
23) $acbdcabdcadb$
24) $acbdcadbacbd$
25) $acbdcadbacdb$
26) $acbdcadbcabd$
27) $acbdcadbcadb$
28) $acdbacbdcabd$
29) $acdbacbdcadb$
30) $acdbacdbcabd$
31) $acdbcabdacbd$
32) $acdbcabdacdb$
33) $acdbcabdcabd$
34) $acdbcabdcadb$
35) $acdbcadbacbd$
36) $acdbcadbcabd$
37) $bacbdacbdcad$
38) $bacbdacdbcad$
39) $bacbdcabdacd$

40) *bacbdcabdcad*
41) *bacbdcadbacd*
42) *bacbdcadbcad*
43) *bacdbacbdcad*
44) *bacdbcabdacd*
45) *bacdbcabdcad*
46) *bdacbdacdbca*
47) *bdacbdcadbac*
48) *bdacbdcadbca*
49) *bdacdbacbdca*
50) *bdacdbacdbca*
51) *bdacdbcabdac*
52) *bdacdbcabdca*
53) *bdacdbcadbac*
54) *bdacdbcadbca*
55) *bdcabdacdbac*
56) *bdcabdacdbca*
57) *bdcabdcadbac*
58) *bdcadbacbdac*
59) *bdcadbacbdca*
60) *bdcadbacdbac*
61) *bdcadbacdbca*
62) *bdcadbcabdac*
63) *bdcadbcadbac*
64) *bcabdacbdacd*
65) *bcabdacbdcad*
66) *bcabdacdbacd*
67) *bcabdacdbcad*
68) *bcabdcabdacd*
69) *bcabdcadbacd*
70) *bcadbacbdacd*
71) *bcadbacbdcad*
72) *bcadbcabdacd*
73) *dacbdacdbcab*
74) *dacbdcadbacb*
75) *dacbdcadbcab*
76) *dacdbacbdcab*
77) *dacdbacdbcab*
78) *dacdbcabdacb*
79) *dacdbcabdcab*
80) *dacdbcadbacb*
81) *dacdbcadbcab*
82) *dbacbdacbdca*
83) *dbacbdacdbca*
84) *dbacbdcabdac*
85) *dbacbdcabdca*
86) *dbacbdcadbac*
87) *dbacbdcadbca*
88) *dbacdbacbdca*
89) *dbacdbcabdac*
90) *dbacdbcabdca*
91) *dbcabdacbdac*
92) *dbcabdacbdca*
93) *dbcabdacdbac*
94) *dbcabdacdbca*
95) *dbcabdcabdac*
96) *dbcabdcadbac*
97) *dbcadbacbdac*
98) *dbcadbacbdca*
99) *dbcadbcabdac*
100) *dcabdacdbacb*
101) *dcabdacdbcab*
102) *dcabdcadbacb*
103) *dcadbacbdacb*
104) *dcadbacbdcab*
105) *dcadbacdbacb*
106) *dcadbacdbcab*
107) *dcadbcabdacb*
108) *dcadbcadbacb*
109) *cabdacbdacdb*
110) *cabdacbdcadb*
111) *cabdacdbacbd*
112) *cabdacdbacdb*
113) *cabdacdbcabd*
114) *cabdacdbcadb*
115) *cabdcabdacdb*
116) *cabdcadbacbd*
117) *cabdcadbacdb*
118) *cadbacbdacbd*
119) *cadbacbdacdb*
120) *cadbacbdcabd*
121) *cadbacbdcadb*
122) *cadbacdbacbd*
123) *cadbacdbcabd*
124) *cadbcabdacbd*
125) *cadbcabdacdb*
126) *cadbcadbacbd*
127) *cbdacbdcadba*
128) *cbdacdbcabda*
129) *cbdacdbcadba*
130) *cbdcabdacdba*
131) *cbdcabdcadba*
132) *cbdcadbacbda*
133) *cbdcadbacdba*
134) *cbdcadbcabda*
135) *cbdcadbcadba*
136) *cdbacbdcabda*
137) *cdbacbdcadba*
138) *cdbacdbcabda*
139) *cdbcabdacbda*
140) *cdbcabdacdba*
141) *cdbcabdcabda*
142) *cdbcabdcadba*
143) *cdbcadbacbda*
144) *cdbcadbcabda*

*Пяткин Артём Валерьевич*
*Сафарова Алиса Исламовна*

Статья поступила
11 марта 2026 г.
После доработки —
10 апреля 2026 г.
Принята к публикации
22 апреля 2026 г.

DISKRETNYI ANALIZ I ISSLEDOVANIE OPERATSII
/DISCRETE ANALYSIS AND OPERATIONS RESEARCH/
April–June 2026. Vol. 33, No. 2. P. 97–114

UDC 519.17 DOI: 10.33048/daio.2026.33.858

# THE FOUR-DIMENSIONAL BOOLEAN CUBE IS NOT 3-REPRESENTABLE

*A. V. Pyatkin*[1, a] and *A. I. Safarova*[2, b]

[1] Sobolev Institute of Mathematics,
4 Koptyug Avenue, 630090 Novosibirsk, Russia
[2] Novosibirsk State University,
2 Pirogov Street, 630090 Novosibirsk, Russia
E-mail: [a] artem@math.nsc.ru, [b] a.safarova1@g.nsu.ru

**Abstract.** A graph is 3-representable if there exists a 3-uniform word whose letters are the vertices of the graph, and two vertices are adjacent if and only if the corresponding letters alternate in the word. In this paper we provide a computer-aided proof of the fact that the 4-dimensional cube is not 3-representable. Illustr. 1, bibliogr. 11.

**Keywords:** word-representable graph, representation number, four-dimensional cube.

## References

1. **S. V. Kitaev** and **A. V. Pyatkin,** On representable graphs, *J. Autom. Lang. Comb.* **13** (1), 45–54 (2008).
2. **S. V. Kitaev** and **A. V. Pyatkin,** Word-representable graphs: A survey, *Diskretn. Anal. Issled. Oper.* **25** (2), 19–53 (2018) [Russian] [*J. Appl. Ind. Math.* **12** (2), 278–296 (2018)].
3. **S. V. Kitaev** and **V. V. Lozin,** *Words and Graphs* (Springer, Cham, 2015).
4. **A. Bouchet,** Circle graph obstructions, *J. Comb. Theory, Ser. B*, **60** (1), 107–144 (1994).
5. **S. V. Kitaev,** On graphs with representation number 3, *J. Autom. Lang. Comb.* **18** (2), 97–112 (2013).
6. **N. S. Alshammari, S. V. Kitaev,** and **A. V. Pyatkin,** On the representation number of grid graphs and cylindric grid graphs (Ithaca, NY, 2025) (e-Print Archive / Cornell Univ., arXiv:2507.16469), DOI: 10.48550/arXiv.2507.16469.

English transl.: *Journal of Applied and Industrial Mathematics* **20** (2) (2026).

7. **T. Dwary, K. Mozhui,** and **K. V. Krishna,** Representation number of word-representable split graphs (Ithaca, NY, 2025) (e-Print Archive / Cornell Univ., arXiv:2502.00872), DOI: 10.48550/arXiv.2502.00872.
8. **B. Broere** and **H. Zantema,** The $k$-dimensional cube is $k$-representable, *J. Autom. Lang. Comb.* **24** (1), 3–12 (2019).
9. **Z. Hefty, P. Horn, C. Muir,** and **A. Owens,** Word-representable graphs: Orientations, posets, and bounds, *Electron. J. Comb.* **31** (4), ID P4.2 (2024).
10. **M. Glen, S. V. Kitaev,** and **A. V. Pyatkin,** On the representation number of a crown graph, *Discrete Appl. Math.* **244**, 89–93 (2018).
11. **Z. Hefty, P. Horn, C. Muir,** and **A. Owens,** Word-representation numbers of graphs: Bottlenecks and bounds, *J. Comb. Theory, Ser. A*, **223**, ID 106215 (2026).

*Artem V. Pyatkin*
*Alisa I. Safarova*

Received March 11, 2026
Revised April 10, 2026
Accepted April 22, 2026