# ПРИКЛАДНАЯ ТЕОРИЯ ГРАФОВ

УДК 519.1, 519.8 DOI 10.17223/20710410/70/3

## О ЦЕЛОЧИСЛЕННОМ ЛИНЕЙНОМ ПРОГРАММИРОВАНИИ ДЛЯ ЗАДАЧ КЛАСТЕРИЗАЦИИ ВЕРШИН ГРАФА[1]

А. В. Моршинин

*Институт математики им. С. Л. Соболева СО РАН, г. Омск, Россия*

**E-mail:** morshinin.alexander@gmail.com

Задачи кластеризации являются важной частью анализа данных. В них требуется разбить заданное множество объектов на несколько подмножеств (кластеров) на основе сходства объектов друг с другом. Задача кластеризации вершин графа является формализацией задачи кластеризации. Сходство объектов задаётся с помощью рёбер некоторого графа, вершины которого взаимно однозначно соответствуют объектам. Существует множество вариантов задачи: с ограничением на число и размер кластеров, взвешенные и ориентированные постановки; все известные варианты задачи являются NP-трудными. Данная работа посвящена одному из подходов к решению задачи — построению моделей целочисленного линейного программирования. Приведён обзор известных и предложены новые подходы к построению таких моделей. Новые модели могут использоваться как для нахождения точных решений, так и для построения приближённых алгоритмов. Проведён вычислительный эксперимент, направленный на оценку времени, необходимого алгоритмам, опирающимся на различные модели, для нахождения точного решения. Показано, что один из алгоритмов, опирающихся на новые модели, быстрее других находит решение для задачи с ограниченным числом кластеров.

**Ключевые слова:** *кластерный граф, целочисленное линейное программирование, NP-трудная задача.*

## ON AN INTEGER LINEAR PROGRAMMING FOR CORRELATION CLUSTERING

A. V. Morshinin

*Sobolev Institute of Mathematics SB RAS, Omsk, Russia*

Clustering problems form an important section of data analysis. In these problems, we need to partition a given set of objects into several subsets (clusters) based on the similarity of the objects to each other. Correlation clustering is a formalization of the clustering problem. Similar objects are connected by edges of a graph and vertices are in one-to-one correspondence with the objects. The problem has several variants: with limited number and size of clusters, weighted, and directed. All known variants are NP-hard. We investigate an approach to solve the problems that involves

[1]Исследование выполнено за счёт гранта РНФ № 22-71-10015.

building integer linear programming models. We review existing models and propose new approaches to model building. The new models can be used both to find exact solutions and to construct approximate algorithms. An experimental study has been conducted to evaluate the computation time to find exact solutions by algorithms based on different models. It showed that one of the algorithms based on the new models is faster than others in finding solutions for a variant of the problem with a limited number of clusters.

**Keywords:** *cluster graph, integer linear programming, NP-hard problem.*

## Введение

В *задаче кластеризации* требуется разбить заданное множество объектов на подмножества (кластеры) на основе сходства между объектами. Одной из наиболее наглядных формализаций этой задачи является *задача кластеризации вершин графа* [1] (*correlation clustering* [2], *cluster editing* [3, 4] и др.), где сходство объектов задаётся рёбрами графа, вершины которого взаимно однозначно соответствуют объектам.

Будем рассматривать только неориентированные графы без петель и кратных рёбер, то есть *обыкновенные графы*. Обыкновенный граф $G = (V, E)$ называется *кластерным*, если каждая его компонента связности является полным графом.

Если $G_1 = (V, E_1)$ и $G_2 = (V, E_2)$ — помеченные графы на одном и том же множестве вершин $V$, то *расстояние* $\mathrm{d}(G_1, G_2)$ между ними определяется как

$$\mathrm{d}(G_1, G_2) = |E_1 \setminus E_2| + |E_2 \setminus E_1|,$$

то есть $\mathrm{d}(G_1, G_2)$ равно числу различающихся рёбер в графах $G_1$ и $G_2$.

Определим следующие множества кластерных графов:

1) $\mathbf{CGS}(V)$ (Cluster Graph Set) — множество всех кластерных графов на множестве вершин $V$;
2) $\mathbf{CGS}_k(V)$ — множество всех кластерных графов на $V$, имеющих $k$ компонент связности $(1 \leqslant k \leqslant |V|)$;
3) $\mathbf{CGS}_{\leqslant k}(V)$ — множество всех кластерных графов на $V$, имеющих не более $k$ компонент связности $(1 \leqslant k \leqslant |V|)$. Очевидно, что $\mathbf{CGS}_{\leqslant k}(V) = \bigcup\limits_{i=1}^{k} \mathbf{CGS}_i(V)$.

Эти множества тесно связаны со следующими минимизационными вариантами задачи кластеризации вершин графа:

— MIN-DISAGREE. Для произвольного графа $G = (V, E)$ найти ближайший к $G$ кластерный граф $C^* \in \mathbf{CGS}(V)$, то есть граф, для которого величина $\mathrm{d}(G, C^*)$ минимальна среди всех графов из $\mathbf{CGS}(V)$.
— MIN-DISAGREE$_k$. Для произвольного графа $G = (V, E)$ и целого числа $k$, $2 \leqslant k \leqslant |V|$, найти ближайший к $G$ кластерный граф $C^* \in \mathbf{CGS}_k(V)$.
— MIN-DISAGREE$_{\leqslant k}$ формулируется аналогично.

Изучение задач кластеризации вершин графа имеет множество приложений. P. Solé и T. Zaslavsky [5] показали связь этих задач с теорией кодирования; R. Shamir, R. Sharan и D. Tsur [3], а также A. Ben-Dor, R. Shamir и Z. Yakhimi [4] — с вычислительной биологией. Изучая задачи классификации документов, N. Bansal, A. Blum и S. Chawla [2] фактически переоткрыли эти задачи. Кластеризация вершин графа связана с такими проблемами, как кластеризация многомерных данных [6], бикластеризация [7] и др.

Вычислительная сложность задач кластеризации вершин графа долгое время оставалась неизвестной. В 1986 г. M. Křivánek и J. Morávek [8] доказали, что задача MIN-DISAGREE является NP-трудной, однако их работа осталась незамеченной. В 2004 г. N. Bansal, A. Blum и S. Chawla [2] и независимо R. Shamir, R. Sharan и D. Tsur [3] показали NP-трудность задачи MIN-DISAGREE. В [2] также доказано, что задача MIN-DISAGREE$_k$ является NP-трудной при любом фиксированном $k \geqslant 2$; в 2006 г. I. Giotis и V. Guruswami [9] опубликовали более простое доказательство этого результата. В том же году А. А. Агеев, В. П. Ильев, А. В. Кононов и А. С. Талевнин [10] доказали, что задачи MIN-DISAGREE$_2$ и MIN-DISAGREE$_{\leqslant 2}$ NP-трудны уже на кубических графах, откуда вывели, что все упомянутые ранее задачи кластеризации вершин графа являются NP-трудными, включая задачу MIN-DISAGREE$_{\leqslant k}$.

Известно множество приближённых алгоритмов для задач кластеризации вершин графа. В [2] представлен 3-приближённый алгоритм для задачи MIN-DISAGREE$_{\leqslant k}$; в [10] доказано существование рандомизированной полиномиальной приближённой схемы для задачи MIN-DISAGREE$_{\leqslant 2}$, а в [9] предложена рандомизированная полиномиальная приближённая схема для задачи MIN-DISAGREE$_{\leqslant k}$ (для любого фиксированного $k \geqslant 2$). Указав, что сложность схемы из [9] лишает её перспективы практического применения, T. Coleman, J. Saunderson и A. Wirth [11] в 2008 г. разработали 2-приближённый алгоритм решения задачи MIN-DISAGREE$_{\leqslant 2}$, применив процедуру локального поиска к каждому допустимому решению, полученному с помощью 3-приближённого алгоритма из [2]. Для задачи MIN-DISAGREE$_2$ В. П. Ильев, С. Д. Ильева и А. А. Навроцкая [12] в 2011 г. предложили 3-приближённый алгоритм, а в 2020 г. В. П. Ильев, С. Д. Ильева и А. В. Моршинин [13] усилили этот результат, предложив 2-приближённый алгоритм. Эти же авторы в [14] представили два 6-приближённых алгоритма для задачи MIN-DISAGREE$_{\leqslant 3}$.

Что касается задачи MIN-DISAGREE, то в 2005 г. M. Charikar, V. Guruswami и A. Wirth [15] показали, что она является APX-трудной, и разработали 4-приближённый алгоритм её решения, опирающийся на *модель целочисленного линейного программирования* (ЦЛП). В 2008 г. N. Ailon, M. Charikar и A. Newman [16] предложили 2,5-приближённый алгоритм для задачи MIN-DISAGREE. В 2015 г. S. Chawla, K. Makarychev, T. Schramm и G. Yaroslavtsev [17] разработали для этой задачи 2,06-приближённый алгоритм.

Настоящая работа посвящена построению моделей ЦЛП для задач кластеризации вершин графа. В п. 1 приводится обзор известного подхода к построению моделей ЦЛП для рассматриваемых задач и предлагается простой способ сокращения количества переменных и ограничений модели. В п. 2 описаны новые подходы к построению моделей ЦЛП, показано, что один из них позволяет значительно сократить количество ограничений для варианта задачи с ограниченным числом кластеров. В п. 3 приведены результаты экспериментального исследования времени работы точных алгоритмов, опирающихся на известные и новые модели ЦЛП.

## 1. Известные модели ЦЛП для задач кластеризации вершин графа

### 1.1. Характеризация множеств **CGS**, **CGS**$_k$ и **CGS**$_{\leqslant k}$ запрещёнными графами

Модели ЦЛП, представленные здесь, опираются на характеризацию кластерных графов запрещёнными графами. Множество кластерных графов может быть описано конечным множеством *запрещённых графов*, которые не могут содержаться в качестве (порождённых) подграфов ни в одном из графов данного множества.

Известно, что граф $G = (V, E)$ принадлежит множеству $\mathbf{CGS}(V)$ тогда и только тогда, когда он не содержит в качестве порождённого подграфа простую цепь $P_2$ [18]. Легко проверить, что граф $G = (V, E)$ принадлежит множеству $\mathbf{CGS}_{\leqslant k}(V)$ тогда и только тогда, когда он принадлежит множеству $\mathbf{CGS}(V)$ и не содержит в качестве порождённого подграфа пустой граф $O_{k+1}$.

Сложнее определить принадлежность графа к множеству $\mathbf{CGS}_k(V)$. Помимо ограничения количества кластеров сверху, необходимо также ограничить их количество снизу. Используем следующее определение: *звёздный лес* $SF$ графа $G = (V, E)$ — это остовный подграф графа $G$, каждая компонента связности которого является звездой [19] (рис. 1); $SF_k$ обозначает звёздный лес, содержащий ровно $k$ звёзд.

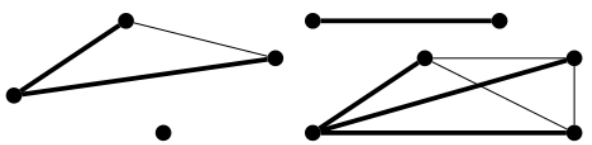

Рис. 1. Звёздный лес графа $G$ выделен жирным. Очевидно, что $G \notin \mathbf{CGS}_i(V), i = 1, 2, 3$

**Утверждение 1.** Граф $G = (V, E)$ принадлежит множеству $\mathbf{CGS}_k(V)$ тогда и только тогда, когда:

1) $G \in \mathbf{CGS}_{\leqslant k}(V)$;
2) $G$ не содержит в качестве подграфа звёздный лес $SF_{k-1}$.

***Доказательство.***

Необходимость. Пусть $G \in \mathbf{CGS}_k(V)$. Очевидно, что $G \in \mathbf{CGS}_{\leqslant k}(V)$. Поскольку $G$ имеет ровно $k$ компонент связности и каждая компонента содержит как минимум одну звезду в качестве остовного подграфа, то минимальный звёздный лес в $G$ состоит ровно из $k$ звёзд.

Достаточность. Пусть $G \in \mathbf{CGS}_{\leqslant k}(V)$ и $G$ не содержит в качестве подграфа звёздный лес $SF_{k-1}$. По определению множества $\mathbf{CGS}_{\leqslant k}(V)$, граф $G$ принадлежит $\mathbf{CGS}_j(V)$ для некоторого $1 \leqslant j \leqslant k$. Предположим, что $j < k$. Тогда, как доказано ранее, $G$ содержит некоторый звёздный лес $SF_j$. Если $j = k - 1$, то $G$ содержит звёздный лес $SF_{k-1}$. Пусть $j < k - 1$. Поскольку пустой граф $O_1$ является звездой, сделаем следующую процедуру: в $SF_j$ возьмём любую вершину степени 1 и удалим ребро, одним из концов которого является эта вершина. Полученный таким образом граф будет звёздным лесом $G$ с $j + 1$ звездой. Последовательно повторяя процедуру $k - 1 - j > 0$ раз, можно построить звёздный лес $SF_{k-1}$. Полученное противоречие доказывает, что $k = j$, а значит, $G \in \mathbf{CGS}_k(V)$.

Утверждение 1 доказано. ■

### 1.2. Модели ЦЛП для задачи MIN-DISAGREE

Рассмотрим произвольный граф $G = (V, E)$ с $n$ вершинами. В [15] предложена модель ЦЛП для задачи MIN-DISAGREE. Кластеризация вершин может быть представлена с помощью бинарных переменных $x_{ij}$, определённых для всех пар вершин $i$ и $j$, где

$$x_{ij} = \begin{cases} 0, & \text{если вершины } i \text{ и } j \text{ принадлежат одному кластеру,} \\ 1 & \text{иначе.} \end{cases}$$

По умолчанию будем считать, что $x_{ii} = 0$. Из транзитивности отношения принадлежности к кластеру следует, что если $x_{ij} = 0$ и $x_{jr} = 0$, то $x_{ir} = 0$. Это свойство обеспечивается неравенствами треугольника:

$$x_{ir} \leqslant x_{ij} + x_{jr} \quad \text{для всех } i, j, r \in V.$$

Эти неравенства гарантируют, что результирующий кластерный граф не содержит простую цепь $P_2$ в качестве порожденного подграфа. D. F. Wahid и E. Hassini в литературном обзоре задач кластеризации [20] в явном виде записали ограничение, отражающее неориентированность исходного графа:

$$x_{ij} = x_{ji} \quad \text{для всех } i, j \in V.$$

Таким образом, получаем следующую модель ЦЛП для задачи MIN-DISAGREE:

$$\sum_{ij \in E} x_{ij} + \sum_{ij \notin E} (1 - x_{ij}) \to \min; \tag{1}$$

$$x_{ir} \leqslant x_{ij} + x_{jr} \quad \text{для всех } i, j, r \in V; \tag{2}$$

$$x_{ij} = x_{ji} \quad \text{для всех } i, j \in V; \tag{3}$$

$$x_{ij} \in \{0, 1\} \quad \text{для всех } i, j \in V. \tag{4}$$

Легко видеть, что для каждой пары вершин необходимо две симметричные переменные, для каждой тройки вершин — шесть неравенств треугольника. Чтобы избавиться от симметричных переменных, запишем вместо неравенства треугольника для каждой упорядоченной тройки вершин три неравенства треугольника для каждой неупорядоченной тройки вершин:

$$\sum_{ij \in E} x_{ij} + \sum_{ij \notin E} (1 - x_{ij}) \to \min; \tag{5}$$

$$\begin{cases} x_{ir} \leqslant x_{ij} + x_{jr} \\ x_{ij} \leqslant x_{ir} + x_{jr} \\ x_{jr} \leqslant x_{ij} + x_{ir} \end{cases} \quad \text{для всех } i, j, r \in V; \tag{6}$$

$$x_{ij} \in \{0, 1\} \quad \text{для всех } i, j \in V. \tag{7}$$

Обе модели содержат $O(n^2)$ переменных и $O(n^3)$ ограничений, но модель (5)–(7) содержит в 2 раза меньше переменных и ограничений, чем модель (1)–(4).

### 1.3. Модели ЦЛП для задачи MIN-DISAGREE$_{\leqslant k}$ и MIN-DISAGREE$_k$

Для построения модели ЦЛП для задачи MIN-DISAGREE$_{\leqslant k}$ достаточно дополнить базовые модели (1)–(4) и (5)–(7) следующим ограничением:

$$x_{i_1 i_2} + \ldots + x_{i_k i_{k+1}} \leqslant (k+2)(k-1)/2 \quad \text{для всех } i_1, \ldots, i_{k+1} \in V. \tag{8}$$

Это неравенство гарантирует, что в любом подмножестве из $(k+1)$ вершин хотя бы одна пара принадлежит одному кластеру, что исключает появление пустого подграфа $O_{k+1}$.

Обе расширенные модели (1)–(4),(8) и (5)–(8) содержат $O(n^{k+1})$ ограничений типа (8). Однако модель (1)–(4),(8) требует примерно в $(k+1)!$ раз больше таких ограничений, чем модель (5)–(8), из-за необходимости учитывать все перестановки вершин.

В задаче MIN-DISAGREE$_k$ необходимо ограничить количество кластеров снизу. Утверждение 1 позволяет легко ввести это ограничение для $k = 2$, а именно: необходимо, чтобы $G$ не содержал в качестве подграфа звёздный лес $SF_1$. Для этого достаточно, чтобы не все вершины графа $G$ были смежны с вершиной 1. Это утверждение можно записать в виде следующего неравенства:

$$\sum_{j=2}^{n} x_{1j} \geqslant 1. \tag{9}$$

Однако для случая $k \geqslant 3$ построение аналогичных компактных линейных ограничений представляет значительную сложность. В п. 2 представлен новый подход к построению моделей ЦЛП для этих задач, который позволяет простым способом ограничить количество кластеров как сверху, так и снизу.

## 2. Новые модели ЦЛП для задач кластеризации вершин графа

### 2.1. Анализ неравенства треугольника

Как было отмечено ранее, для каждой тройки вершин модели (1)–(4) и (5)–(7) содержат соответственно 6 и 3 неравенства треугольника. Рассмотрим следующий подход к сокращению их количества.

Ключевое наблюдение: в моделях (1)–(4) и (5)–(7) неравенство треугольника запрещает появление $P_2$ как порождённого подграфа. Для любой тройки вершин сумме $S = x_{ij} + x_{ir} + x_{jr}$ соответствует (рис. 2):

1) $S = 3$: пустой граф $O_3$;
2) $S = 2$: граф $K_2 \cup O_1$;
3) $S = 1$: простая цепь $P_2$ (запрещенная конфигурация);
4) $S = 0$: полный граф $K_3$ (все вершины в одном кластере).

Рис. 2. Все неизоморфные графы с тремя вершинами

Для исключения запрещённого случая $S = 1$ применим метод линеаризации из работы [21]. Введём условие

$$|x_{ij} + x_{ir} + x_{jr} - 1| \geqslant \varepsilon$$

для некоторого небольшого $\varepsilon > 0$ (например, $\varepsilon = 10^{-3}$).

Используя бинарные переменные $y_{ijr} \in \{0, 1\}$, связывающие две области допустимых значений, и большое число $M$ (например, $M = 10^3$), получаем

$$\sum_{ij \in E} x_{ij} + \sum_{ij \notin E} (1 - x_{ij}) \to \min; \tag{10}$$

$$x_{ij} + x_{ir} + x_{jr} - 1 \geqslant \varepsilon - (1 - y_{ijr})M \quad \text{для всех } i, j, r \in V; \tag{11}$$

$$x_{ij} + x_{ir} + x_{jr} - 1 \leqslant -\varepsilon + y_{ijr}M \quad \text{для всех } i, j, r \in V; \tag{12}$$

$$x_{ij} \in \{0, 1\} \quad \text{для всех } i, j \in V\ , \tag{13}$$

$$y_{ijr} \in \{0, 1\} \quad \text{для всех } i, j, r \in V. \tag{14}$$

Переменная $y_{ijr}$ равна 1, если $x_{ij} + x_{ir} + x_{jr} - 1 \geqslant \varepsilon$, иначе она равна 0.

Для каждой тройки вершин вместо шести неравенств в модели (1)–(4) и трёх неравенств в модели (5)–(7) имеется два неравенства. Нам пришлось добавить $O(n^3)$ переменных. Эта модель имеет иную структуру области допустимых решений, что является хорошим аргументом для её дальнейшего теоретического исследования (например, для построения приближённых алгоритмов). Однако данный подход не решает фундаментальную проблему поиска точных решений: количество неравенств (8) всё ещё быстро растёт. Это требует разработки новых методов построения моделей ЦЛП.

### 2.2. Модели ЦЛП для задач MIN-DISAGREE$_{\leqslant 2}$ и MIN-DISAGREE$_2$

Идея работы с абсолютными величинами, применённая для модели (10)–(14), может быть эффективно использована для построения моделей ЦЛП другого типа. Рассмотрим задачу MIN-DISAGREE$_{\leqslant 2}$.

Для каждой вершины $i \in V$ введём бинарную переменную $x_i$:

$$x_i = \begin{cases} 0, & \text{если вершина } i \text{ принадлежит первому кластеру,} \\ 1, & \text{если вершина } i \text{ принадлежит второму кластеру.} \end{cases}$$

Заметим, что если для вершин $i, j \in V$ в графе $G$ существует ребро $ij \in E$, то выражение $|x_i - x_j|$ равно 0 при $x_i = x_j$, иначе оно равно 1. Если для этих вершин не существует ребра $ij \notin E$, то выражение $|x_i + x_j - 1|$ равно 0 при $x_i \neq x_j$, иначе оно равно 1. Используем данный факт для построения модели целочисленного программирования (ЦП):

$$\sum_{ij \in E} |x_i - x_j| + \sum_{ij \notin E} |x_i + x_j - 1| \to \min,$$
$$x_i \in \{0, 1\} \text{ для всех } i \in V.$$

Эта модель не является линейной. Применим известный приём, который сделает её линейной [22]. Представим каждый модуль в целевой функции в виде суммы двух бинарных переменных $u_{ij} + v_{ij}$, а выражение под модулем — в виде разности этих переменных $v_{ij} - u_{ij}$ $(u_{ij} \cdot v_{ij} = 0)$:

$$\sum_{i,j \in V} u_{ij} + v_{ij} \to \min; \tag{15}$$
$$x_i - x_j + u_{ij} - v_{ij} = 0 \text{ для всех } i, j \in V,\ ij \in E; \tag{16}$$
$$x_i + x_j - 1 + u_{ij} - v_{ij} = 0 \text{ для всех } i, j \in V,\ ij \notin E; \tag{17}$$
$$x_i \in \{0, 1\} \text{ для всех } i \in V; \tag{18}$$
$$u_{ij} \in \{0, 1\} \text{ для всех } i, j \in V; \tag{19}$$
$$v_{ij} \in \{0, 1\} \text{ для всех } i, j \in V. \tag{20}$$

Эта модель содержит $O(n^2)$ переменных и $O(n^2)$ ограничений. Легко получить модель ЦЛП для задачи MIN-DISAGREE$_2$, добавив всего два ограничения. Они гарантируют, что не все вершины одновременно принадлежат одному кластеру:

$$\sum_{i \in V} x_i \geqslant 1; \tag{21}$$
$$\sum_{i \in V} x_i \leqslant n - 1. \tag{22}$$

### 2.3. Модели ЦЛП для задач MIN-DISAGREE$_{\leqslant k}$ и MIN-DISAGREE$_k$

Для задач с ограничением на число кластеров при $k \geqslant 3$ нельзя использовать одну бинарную переменную для описания принадлежности вершин к кластерам. Вместо этого используем *унитарный код*. Для каждой вершины $i \in V$ и кластера $r \in \{1, \ldots, k\}$ введем бинарную переменную $x_{ir}$:

$$x_{ir} = \begin{cases} 1, & \text{если вершина } i \text{ принадлежит кластеру } r, \\ 0 & \text{иначе.} \end{cases}$$

Таким образом, для каждой вершины $i \in V$ существует унитарный код $(x_{i1}, \ldots, x_{ik})$, в котором лишь одна координата равна 1.

Заметим, что при таком кодировании если для вершин $i, j \in V$ существует ребро $ij \in E$, то сумма $\frac{1}{2}\sum\limits_{r=1}^{k} |x_{ir} - x_{jr}|$ равна 0 тогда и только тогда, когда вершины $i$ и $j$ принадлежат одному кластеру, иначе она равна 1. С другой стороны, если $ij \notin E$, то значение $\frac{1}{2}\left(\sum\limits_{r=1}^{k} |x_{ir} + x_{jr} - 1| - k + 2\right)$ равно 0 тогда и только тогда, когда вершины $i$ и $j$ принадлежат разным кластерам, иначе оно равно 1.

Действительно, если вершины $i$ и $j$ принадлежат одному кластеру, то им соответствуют одинаковые унитарные коды, если разным, то их унитарные коды отличаются в двух координатах. В первом случае, покомпонентно вычитая векторы и беря полученную разность по модулю, мы либо получим вектор из 0 ($i$ и $j$ принадлежат одному кластеру), либо вектор только с двумя 1 ($i$ и $j$ принадлежат разным кластерам). Аналогично во втором случае: мы либо получим вектор из 1 ($i$ и $j$ принадлежат одному кластеру), либо вектор только с двумя 0 ($i$ и $j$ принадлежат разным кластерам). Дальнейшие вычисления тривиальны.

Используя замену, аналогичную замене для модели (15)–(20), мы получаем следующую модель ЦЛП для задачи MIN-DISAGREE$_{\leqslant k}$:

$$\sum_{i,j \in V} \sum_{r=1}^{k} u_{ijr} + v_{ijr} \to \min; \tag{23}$$

$$x_{ir} - x_{jr} + u_{ijr} - v_{ijr} = 0 \quad \text{для всех } i, j \in V,\ ij \in E,\ r \in \{1, \ldots, k\}; \tag{24}$$

$$x_{ir} + x_{jr} - 1 + u_{ijr} - v_{ijr} = 0 \quad \text{для всех } i, j \in V,\ ij \notin E,\ r \in \{1, \ldots, k\}; \tag{25}$$

$$\sum_{r=1}^{k} x_{ir} = 1 \quad \text{для всех } i \in V,\ r \in \{1, \ldots, k\}; \tag{26}$$

$$x_{ir} \in \{0, 1\} \quad \text{для всех } i \in V,\ r \in \{1, \ldots, k\}; \tag{27}$$

$$u_{ijr} \in \{0, 1\} \quad \text{для всех } i, j \in V,\ r \in \{1, \ldots, k\}; \tag{28}$$

$$v_{ijr} \in \{0, 1\} \quad \text{для всех } i, j \in V,\ r \in \{1, \ldots, k\}. \tag{29}$$

Равенство (26) означает, что одна вершина может принадлежать одному кластеру.

В отличие от моделей (1)–(4),(8), (5)–(8) и (10)–(14),(8), легко построить модель ЦЛП для задачи MIN-DISAGREE$_k$. Для этого достаточно добавить неравенства, гарантирующие, что в каждом кластере есть хотя бы одна вершина:

$$\sum_{i \in V} x_{ir} \geqslant 1 \quad \text{для всех } r \in \{1, \ldots, k\}. \tag{30}$$

Заметим, что модели (23)–(29) и (23)–(30) содержат $O(kn^2)$ переменных и $O(kn^2)$ ограничений. Таким образом, полученные модели имеют значительно меньше ограничений, хотя количество переменных в $O(k)$ раз больше.

## 3. Экспериментальное исследование

### 3.1. Описание вычислительного эксперимента

Все перечисленные модели ЦЛП могут быть интересны для построения приближённых алгоритмов, которые могут стать объектом дальнейшего исследования. Мы же сосредоточимся на исследовании времени работы точных алгоритмов, опирающихся на описанные модели, при нахождении оптимальных решений на графах малой размерности. Такое исследование интересно по двум причинам. Во-первых, существуют практические задачи, в которых количество объектов не может быть слишком велико. Таковой является, например, задача разделения небольшой социальной группы (рабочий коллектив, школьный класс и др.) на подгруппы, максимизирующая взаимную симпатию внутри подгрупп [23]. Очевидно, что предпочтительно уметь находить оптимальное решение для групп как можно большего размера. Во-вторых, нахождение точных решений позволяет проводить предварительный анализ приближённых алгоритмов, направленный на построение статистических оценок точности. Здесь также чем большую задачу можно решить оптимально, тем качественнее будет предварительный анализ. При этом для нахождения «хороших» допустимых решений на графах большей размерности можно использовать приближённые алгоритмы, описанные во введении.

Для сравнения времени работы точных алгоритмов, опирающихся на различные модели ЦЛП, проведён вычислительный эксперимент. Цель эксперимента заключалась в том, чтобы на основе статистических данных сравнить различные точные алгоритмы между собой и выявить лучший из них для каждой из задач. Все описанные модели реализованы с помощью языка программирования Python и его библиотеки Python-MIP. В качестве решателя выбран IBM ILOG CPLEX. Вычисления производились на NEOS Server с четырьмя ядрами центрального процессора. Выбор библиотеки Python-MIP мотивирован простотой записи реализованной модели в файл MPS-формата, который отправлялся на NEOS Server. Весь необходимый код доступен по ссылке `github.com/BIGADIL/graph_correlation_clustering_mip`.

Опишем схему проведения вычислительного эксперимента:

1) вводится вероятностное распределение на множестве входов исследуемой задачи, то есть задаётся вероятностное пространство на множестве графов;
2) в соответствии с вероятностным распределением проводится случайный выбор $N$ графов, на которых исследуемыми алгоритмами решается задача;
3) для каждого полученного решения вычисляется время работы, которое является случайной величиной;
4) на основе статистических данных вычисляются оценка математического ожидания времени работы и его доверительный интервал для исследуемых алгоритмов.

Итак, введём на множестве всех графов вероятностное распределение. Для этого зафиксируем параметр $p \in (0, 1)$. Случайный $n$-вершинный граф $G = (V, E)$ будем получать с использованием следующей процедуры. Для каждой пары вершин $(u, v)$ проводится независимый случайный эксперимент, исходами которого будут наличие ребра $uv$ с вероятностью $p$ и отсутствие ребра с вероятностью $1 - p$. Таким образом,

граф $G$ можно рассматривать как случайный вектор, каждая координата которого соответствует паре вершин графа $G$. Семейство $n$-вершинных графов с введённым таким образом распределением обозначается $G(n,p)$ и используется как при теоретическом изучении графов, так и в экспериментальных исследованиях (модель Эрдеша — Реньи) [24].

Параметр $p$ в вероятностной модели представляет собой математическое ожидание плотности случайного графа $G=(V,E)$, которая определяется как $2|E|/n(n-1)$. В экспериментах использовались значения $p$ из множества $\{0{,}33,\ 0{,}5,\ 0{,}67\}$.

Рассмотрим теперь параметр $n$ семейства $G(n,p)$. Главным ограничением на количество вершин в тестовых задачах была сложность отыскания оптимального значения целевой функции. В рамках эксперимента для каждой пары значений $n$ и $p$ было решено по 100 задач. Если для какого-то значения $n$ из 100 тестовых задач исследуемый алгоритм не справлялся хотя бы с 3 % задач за 6500 с, такое значение $n$ не участвовало в исследовании. Также соблюдалось ограничение NEOS Server на размер модели, равное 16,5 Мбайт. Таким образом, для каждого алгоритма и для каждого значения $p$ было подобрано своё граничное значение $n$.

В качестве оценки математического ожидания времени работы алгоритма при заданных значениях параметров $n$ и $p$ взято его среднее время работы по серии задач при этих параметрах. Поскольку распределение времени работы чаще всего не является нормальным, построение доверительного интервала среднего производилось с помощью процедуры *бутстрепа*. Опишем простейшую реализацию этой процедуры. Из имеющейся выборки генерируется $B$ псевдовыборок того же размера, что и исходная, методом случайного выбора с возвращением. Для каждой псевдовыборки вычисляется псевдостатистика среднего времени работы. После этого псевдостатистики сортируются в порядке возрастания. На уровне значимости $\alpha$ слева и справа отбрасывается по $[\alpha B/2]$ элементов. Среди оставшихся крайние левый и правый элементы являются границами доверительного интервала среднего времени работы. В рамках эксперимента использовалась реализация *метода бутстрепа с коррекцией смещения и ускорением* (BCa) из библиотеки SciPy со значениями параметров $B=10000$ и $\alpha=0{,}05$. Более подробно о процедуре бутстрепа и BCa можно прочитать в [25, 26].

Для исследования выбраны задачи MIN-DISAGREE$_{\leqslant k}$ при $k=2,3$ и MIN-DISAGREE; задача MIN-DISAGREE$_k$ не изучалась. Такой выбор обусловлен следующими факторами:

1) выбранные задачи являются наиболее изученными;
2) для задачи MIN-DISAGREE$_2$ к моделям (1)–(4),(8), (5)–(8) и (10)–(14),(8) необходимо добавить одно ограничение (9), а к модели (15)–(20) — два ограничения (21) и (22), что значительно меньше количества других ограничений. Это дополнение незначительно влияет на общее время решения;
3) для задачи MIN-DISAGREE$_k$, $k\geqslant 3$, к модели (23)–(29) необходимо добавить $k$ ограничений (30), что значительно меньше количества других ограничений. Эти дополнения незначительно влияют на общее время решения. К тому же не существует альтернативы для модели (23)–(30), с которой её можно было бы сравнить.

Модель (1)–(4) эквивалентна модели (5)–(7), но содержит больше переменных и ограничений. Результаты разведочного анализа показали, что алгоритмы, опирающиеся на модель (1)–(4) и её производные, требуют значительно больше времени для достижения оптимума по сравнению с алгоритмами, опирающимися на модель (5)–(7)

и её производные. Так, для задачи MIN-DISAGREE$_{\leqslant 2}$ при значении параметров $n = 35$ и $p = 0{,}33$ точный алгоритм, опирающийся на модель (1)–(4),(8), находит оптимальное решение в среднем за 2807,1 с. При тех же параметрах точный алгоритм, опирающийся на модель (5)–(8), находит точное решение в среднем за 123,1 с, что более чем в 22 раза быстрее. На основании этих результатов было принято решение исключить модель (1)–(4) и её производные из экспериментального исследования.

### 3.2. Экспериментальное исследование алгоритмов для задачи MIN-DISAGREE$_{\leqslant k}$

Начнём со случая $k = 2$. Обозначим через **TR**, **IN** и **MOD** точные алгоритмы, опирающиеся на модели (5)–(8), (10)–(14),(8) и (15)–(20) соответственно.

Табл. 1 содержит размер (в мегабайтах) каждой из моделей в зависимости от количества вершин. Видно, что модели (5)–(8) и (10)–(14),(8) имеют примерно одинаковый размер и превышают выделенные 16,5 Мбайт при $n = 55$. В свою очередь, модель (15)–(20) остаётся компактной до $n = 95$ включительно. Тем не менее дополнительные ограничения могут сужать область допустимых решений, ускоряя поиск оптимального решения.

Таблица 1

**Размер моделей ЦЛП для задачи MIN-DISAGREE$_{\leqslant 2}$**

| $n$ | (5)–(8) | (15)–(20) | (10)–(14),(8) |
|---|---|---|---|
| 20 | 0,56 | 0,04 | 0,60 |
| 25 | 1,13 | 0,07 | 1,23 |
| 30 | 2,02 | 0,10 | 2,18 |
| 35 | 3,29 | 0,15 | 3,55 |
| 40 | 4,98 | 0,20 | 5,40 |
| 45 | 7,17 | 0,25 | 7,79 |
| 50 | 9,94 | 0,32 | 10,81 |
| 55 | 13,36 | 0,39 | 14,50 |
| 60 | 17,61 | 0,46 | 18,97 |
| 65 | 22,61 | 0,55 | 24,37 |
| 70 | 28,46 | 0,64 | 30,73 |
| 75 | 35,23 | 0,74 | 38,05 |
| 80 | 42,97 | 0,84 | 46,47 |
| 85 | 51,75 | 0,96 | 55,99 |
| 90 | 61,65 | 1,07 | 66,80 |
| 95 | 72,72 | 1,20 | 78,83 |

Среднее время работы точных алгоритмов сильно отличается при $p = 0{,}33\,/\,0{,}5$ и $p = 0{,}67$ (табл. 2–4 и рис. 3).

В случае $p = 0{,}33\,/\,0{,}5$ худшие результаты принадлежат алгоритму **IN**. При $n = 40$ и $p = 0{,}5$ его среднее время работы составляет 3612,7 с, что почти в 4 раза больше, чем у алгоритма **TR**. При $n = 40$ и $p = 0{,}33$ среднее время работы алгоритма **IN** в 1,8 раз больше, чем у алгоритма **TR**, среднее время работы которого равно 812,4 с. Однако оба алгоритма сталкиваются с ограничением по времени при $n = 45$. Лучшие результаты принадлежат алгоритму **MOD**. Лишь при $p = 0{,}33$ и $n = 50$ среднее время его работы приближается к 3600 с и при $n = 55$ сталкивается с ограничением по времени. При $n = 40$ и $p = 0{,}33$ среднее время работы алгоритма **MOD** в 8,4 раз меньше среднего времени работы алгоритма **TR**, а при $n = 40$ и $p = 0{,}5$ — в 5,8 раз. Отметим, что

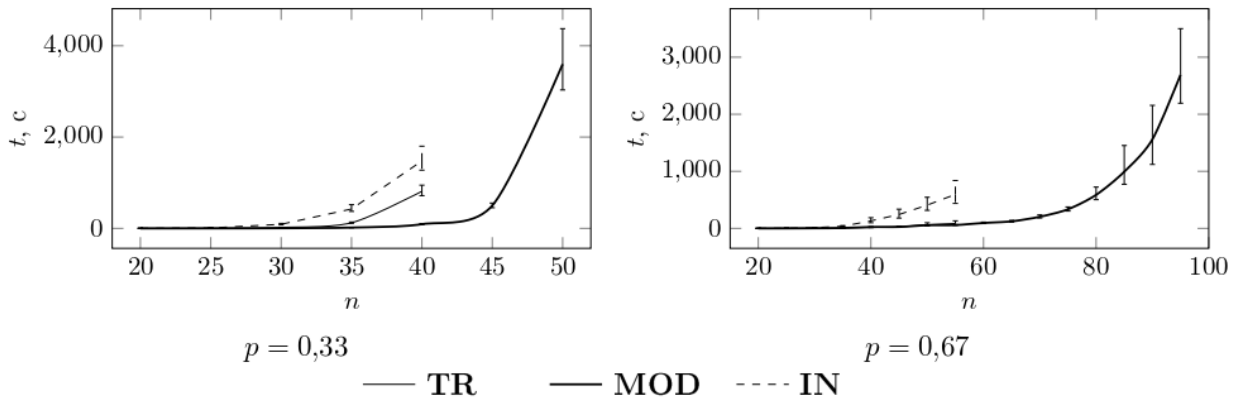


Рис. 3. Среднее время ($t$, с) работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 2}$

доверительные интервалы среднего времени работы всех алгоритмов не пересекаются, а значит, среднее время работы алгоритма **MOD** статистически значимо наименьшее.

Таблица 2

**Среднее время работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 2}$, с**

| $n$ | $p$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0,33 | | | 0,5 | | | 0,67 | | |
| | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** |
| 20 | 0,5 | 0,1 | 1,8 | 0,7 | 0,2 | 1,7 | 0,2 | 0,2 | 0,6 |
| 25 | 3,7 | 1,2 | 9,7 | 5,2 | 2,1 | 12,5 | 0,9 | 0,6 | 2,6 |
| 30 | 21,1 | 4,5 | 89,7 | 43,8 | 9,7 | 179,8 | 3,4 | 3,1 | 13,3 |
| 35 | 123,1 | 18,2 | 424,8 | 240,3 | 36,3 | 1344,9 | 7,3 | 6,7 | 37,4 |
| 40 | 812,4 | 96,2 | 1476,1 | 952,1 | 162,9 | 3612,7 | 16,6 | 24,3 | 131,7 |
| 45 | – | 499,4 | – | – | 623,7 | – | 28,8 | 26,1 | 243,3 |
| 50 | – | 3598,5 | – | – | 2153,2 | – | 63,9 | 53,1 | 407,9 |
| 55 | – | – | – | – | – | – | 85,4 | 58,8 | 595,2 |
| 60 | – | – | – | – | – | – | – | 94,9 | – |
| 65 | – | – | – | – | – | – | – | 123,5 | – |
| 70 | – | – | – | – | – | – | – | 203,8 | – |
| 75 | – | – | – | – | – | – | – | 334,7 | – |
| 80 | – | – | – | – | – | – | – | 587,8 | – |
| 85 | – | – | – | – | – | – | – | 993,1 | – |
| 90 | – | – | – | – | – | – | – | 1560,5 | – |
| 95 | – | – | – | – | – | – | – | 2694,6 | – |

В случае $p = 0{,}67$ худшие показатели также принадлежат алгоритму **IN**. При $n = 55$ среднее время его работы составляет 595,2 с. В свою очередь, алгоритмы **TR** и **MOD** при том же значении $n$ тратят в среднем 85,4 и 58,8 с соответственно. Интересно, что доверительные интервалы среднего времени работы алгоритмов **TR** и **MOD** пересекаются до $n = 55$ включительно. Это не позволяет говорить о статистически значимых различиях. Однако при $n = 60$ занимаемый объём памяти становится препятствием к отысканию оптимальных решений для алгоритмов **TR** и **IN**, в то время как алгоритм **MOD** сталкивается с ограничением по времени при $n = 100$.

Таким образом, можно утверждать, что наилучшим алгоритмом для задачи MIN-DISAGREE$_{\leqslant 2}$ является алгоритм **MOD**. Его среднее время работы при $p = 0{,}33\,/\,0{,}5$ статистически значимо наименьшее. При $p = 0{,}67$ нет статистически значимых отли-

Таблица 3

**Границы доверительного интервала среднего времени работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 2}$, $p = 0{,}33$ и $p = 0{,}5$, с**

| $n$ | $p$ | | | | | |
|---|---|---|---|---|---|---|
| | 0,33 | | | 0,5 | | |
| | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** |
| 20 | [0,4, 0,6] | [0,1, 0,2] | [1,6, 2,1] | [0,6, 0,8] | [0,1, 0,2] | [1,5, 1,9] |
| 25 | [3,4, 4,1] | [1,1, 1,2] | [9,2, 10,3] | [4,6, 5,9] | [1,8, 2,3] | [11,3, 13,9] |
| 30 | [19,6, 22,9] | [4,2, 4,7] | [76,7, 109,2] | [35,5, 54,7] | [8,5, 10,9] | [145,4, 221,7] |
| 35 | [111,8, 138,9] | [16,6, 20,1] | [372,4, 477,2] | [189,1, 303,1] | [30,5, 42,7] | [1030,2, 1776,9] |
| 40 | [717,1, 946,1] | [85,7, 109,5] | [1275,5, 1799,9] | [791,1, 1177,3] | [133,6, 200,3] | [2953,3, 4330,3] |
| 45 | – | [453,2, 554,5] | – | – | [511,4, 767,8] | – |
| 50 | – | [3031,5, 4373,9] | – | – | [1881,5, 2479,4] | – |

Таблица 4

**Границы доверительного интервала среднего времени работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 2}$, $p = 0{,}67$, с**

| $n$ | **TR** | **MOD** | **IN** |
|---|---|---|---|
| 20 | [0,1, 0,3] | [0,1, 0,2] | [0,5, 0,7] |
| 25 | [0,8, 1,2] | [0,5, 0,7] | [2,1, 3,3] |
| 30 | [2,7, 4,4] | [2,7, 3,4] | [10,4, 16,8] |
| 35 | [5,9, 9,4] | [6,1, 7,6] | [29,7, 49,1] |
| 40 | [13,2, 23,8] | [21,1, 28,4] | [98,8, 186,9] |
| 45 | [23,1, 37,6] | [22,9, 29,8] | [181,5, 336,5] |
| 50 | [46,8, 98,2] | [46,8, 60,2] | [316,1, 545,1] |
| 55 | [62,5, 132,7] | [53,9, 67,4] | [437,6, 838,7] |
| 60 | – | [84,2, 110,3] | – |
| 65 | – | [111,3, 140,3] | – |
| 70 | – | [182,3, 236,5] | – |
| 75 | – | [304,1, 375,7] | – |
| 80 | – | [503,2, 724,1] | – |
| 85 | – | [773,2, 1452,2] | – |
| 90 | – | [1121,1, 2000,1] | – |
| 95 | – | [2193,8, 3500,1] | – |

чий в среднем времени работы алгоритмов **TR** и **MOD**, однако последний требует гораздо меньше памяти.

Отметим, что в случае $p = 0{,}33\,/\,0{,}5$ оптимальным решением в 100 % случаев является кластерный граф с двумя кластерами. При $p = 0{,}67$ лишь в 49,5 % случаев оптимальным является граф с двумя кластерами, в оставшихся 51,5 % случаев оптимальный граф содержит один кластер. Это один из факторов, объясняющих лучшее среднее время работы всех алгоритмов на плотных графах.

Перейдём к случаю $k = 3$. В табл. 5 приведены размеры моделей для задачи MIN-DISAGREE$_{\leqslant 3}$ в мегабайтах (Мбайт). Как и в случае $k = 2$, модели (5)–(8) и (10)–(14), (8) имеют примерно одинаковый размер и превышают выделенные 16,5 Мбайт при $n = 40$, модель (23)–(29) занимает немного памяти до $n = 70$ включительно.

Результаты эксперимента представлены в табл. 6–8 и на рис. 4. Худшее среднее время работы в случае $p = 0{,}33\,/\,0{,}5$ принадлежит алгоритму **IN**. При $p = 0{,}5$ и $n = 30$ доверительные интервалы среднего времени работы алгоритмов **IN** и **TR** пересекаются, что не позволяет говорить о статистически значимых отличиях. Однако при

Таблица 5

**Размер моделей ЦЛП для задачи MIN-DISAGREE$_{\leqslant 3}$**

| $n$ | (5)–(8) | (23)–(29) | (10)–(14),(8) |
|---|---|---|---|
| 15 | 0,48 | 0,07 | 0,50 |
| 20 | 1,55 | 0,14 | 1,60 |
| 25 | 3,92 | 0,22 | 4,01 |
| 30 | 8,25 | 0,33 | 8,40 |
| 35 | 15,37 | 0,46 | 15,64 |
| 40 | 26,48 | 0,61 | 26,82 |
| 45 | 42,98 | 0,79 | 43,49 |
| 50 | 66,16 | 0,98 | 66,87 |
| 55 | 97,47 | 1,19 | 98,45 |
| 60 | 138,64 | 1,43 | 139,91 |
| 65 | 191,46 | 1,69 | 193,18 |
| 70 | 258,65 | 1,97 | 260,48 |

$p = 0{,}33$ их доверительные интервалы среднего времени работы не пересекаются, а при $p = 0{,}33$ и $n = 30$ среднее время работы алгоритма **TR** составляет 1039,1 с, что более чем в 3 раза меньше, чем среднее время работы алгоритма **IN**. Оба алгоритма сталкиваются с ограничением по времени при $n = 35$. Вновь лучшие результаты принадлежат алгоритму **MOD**. Его доверительные интервалы среднего времени работы не пересекаются с доверительными интервалами алгоритмов **IN** и **TR**. Среднее время работы алгоритма **MOD** при $p = 0{,}33$ и $n = 30$ в 5 раз меньше, чем у алгоритма **TR**, а при $p = 0{,}5$ и $n = 30$ — почти в 19 раз.

Таблица 6

**Среднее время работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 3}$, с**

| $n$ | $p$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0,33 | | | 0,5 | | | 0,67 | | |
| | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** |
| 15 | 0,1 | 0,4 | 0,7 | 0,2 | 0,3 | 0,7 | 0,1 | 0,2 | 1,3 |
| 20 | 2,6 | 2,6 | 18,6 | 4,7 | 3,2 | 20,1 | 0,4 | 1,8 | 4,4 |
| 25 | 51,8 | 18,1 | 179,9 | 104,1 | 16,5 | 250,9 | 2,8 | 6,2 | 28,1 |
| 30 | 1039,1 | 187,1 | 3282,5 | 4773,7 | 254,9 | 6138,6 | 21,3 | 18,9 | 75,2 |
| 35 | – | 1996,5 | – | – | 2438,8 | – | 127,1 | 61,5 | 600,8 |
| 40 | – | | – | – | – | – | – | 81,6 | – |
| 45 | – | | – | – | – | – | – | 242,3 | – |
| 50 | – | – | – | – | – | – | – | 352,1 | – |
| 55 | – | – | – | – | – | – | – | 488,1 | – |
| 60 | – | – | – | – | – | – | – | 877,9 | – |
| 65 | – | – | – | – | – | – | – | 1631,5 | – |
| 70 | – | – | – | – | – | – | – | 2432,2 | – |

Таблица 7

**Границы доверительного интервала среднего времени работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 3}$, $p = 0{,}33$ и $p = 0{,}5$, с**

| $n$ | $p$ | | | | | |
|---|---|---|---|---|---|---|
| | 0,33 | | | 0,5 | | |
| | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** |
| 15 | [0,1, 0,2] | [0,4, 0,5] | [0,6, 1,0] | [0,2, 0,3] | [0,3, 0,4] | [0,6, 1,0] |
| 20 | [2,3, 3,0] | [2,5, 2,8] | [16,5, 20,9] | [4,1, 5,5] | [3,1, 3,3] | [18,2, 22,1] |
| 25 | [46,1, 59,2] | [16,1, 21,2] | [156,3, 212,6] | [80,3, 142,3] | [15,3, 17,9] | [228,8, 280,0] |
| 30 | [914,4, 1216,7] | [171,5, 205,8] | [2824,7, 3977,8] | [3743,3, 6002,5] | [232,2, 280,6] | [5378,5, 7125,8] |
| 35 | – | [1813,2, 2254,9] | – | – | [2211,5, 2715,7] | – |

Таблица 8

**Границы доверительного интервала среднего времени работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 3}$, $p = 0{,}67$, с**

| $n$ | **TR** | **MOD** | **IN** |
|---|---|---|---|
| 15 | [0,1, 0,2] | [0,2, 0,3] | [1,2, 1,5] |
| 20 | [0,4, 0,6] | [1,6, 1,9] | [3,4, 6,2] |
| 25 | [2,1, 4,1] | [5,7, 6,6] | [20,3, 39,2] |
| 30 | [13,8, 41,4] | [17,3, 21,2] | [45,9, 160,8] |
| 35 | [86,7, 217,4] | [56,2, 72,1] | [413,0, 983,7] |
| 40 | – | [73,8, 92,3] | – |
| 45 | – | [185,1, 426,4] | – |
| 50 | – | [316,1, 412,5] | – |
| 55 | – | [438,4, 567,1] | – |
| 60 | – | [787,5, 994,2] | – |
| 65 | – | [1405,0, 1942,8] | – |
| 70 | – | [2054,9, 2990,1] | – |

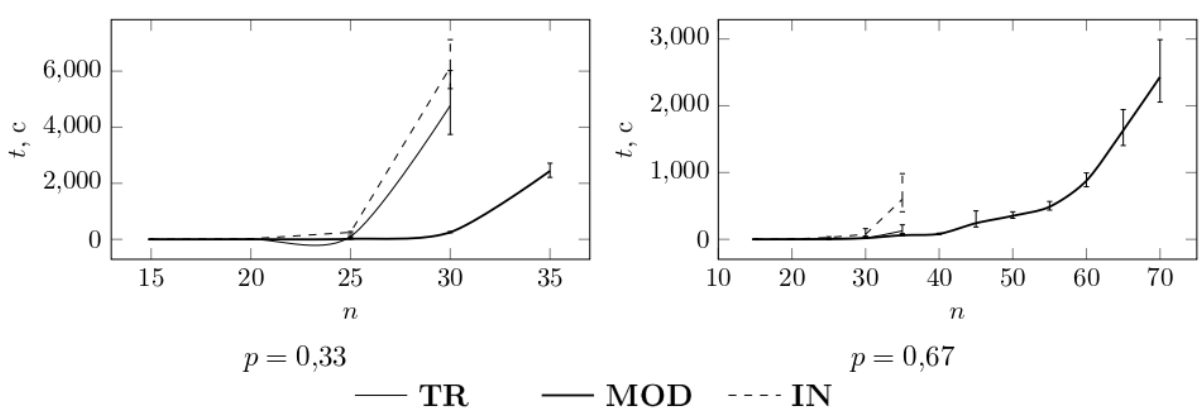


Рис. 4. Среднее время ($t$, с) работы алгоритмов для задачи MIN-DISAGREE$_{\leqslant 3}$

При $p = 0{,}67$ алгоритм **IN** опять показывает себя хуже алгоритмов **TR** и **MOD**. Интересно, что до $n = 25$ алгоритм **TR** имеет лучшее среднее время работы, чем алгоритм **MOD**, причём их доверительные интервалы не пересекаются. При $n = 30$ среднее время работы алгоритма **MOD** становится меньше среднего времени работы алгоритма **TR**, однако их доверительные интервалы пересекаются. При $n = 35$ доверительные интервалы перестают пересекаться и среднее время работы алгоритма **MOD** становится меньше, чем у алгоритма **TR**. При $n = 40$ опорные модели алгорит-

мов **IN** и **TR** исчерпывают память, а алгоритм **MOD** сталкивается с ограничением по времени при $n = 70$.

Заметим, что результаты эксперимента очень похожи на результаты эксперимента при $k = 2$. При $p = 0{,}33\,/\,0{,}5$ оптимальным решением в 99,4 % случаев является граф с тремя кластерами, а в 0,6 % — с двумя. При $p = 0{,}67$ лишь в 7,4 % случаев оптимальное решение содержит три кластера, в оставшихся случаях — один или два кластера (41 и 51,6 % соответственно).

### 3.3. Экспериментальное исследование алгоритмов для задачи MIN-DISAGREE

Если в модели (23)–(29) заменить $k$ на $n$, то получится модель ЦЛП для задачи MIN-DISAGREE. Эта модель содержит $O(n^3)$ переменных и $O(n^3)$ ограничений. Сравним её с другими моделями. Вновь обозначим через **TR**, **IN** и **MOD** точные алгоритмы, опирающиеся на модели (5)–(7), (10)–(14) и (23)–(29) (табл. 9–12).

Таблица 9

**Размер моделей ЦЛП для задачи MIN-DISAGREE, Мбайт**

| $n$ | (5)–(7) | (23)–(29) | (10)–(14) |
|---|---|---|---|
| 15 | 0,15 | 0,38 | 0,17 |
| 20 | 0,39 | 0,96 | 0,44 |
| 25 | 0,79 | 1,96 | 0,90 |
| 30 | 1,42 | 3,47 | 1,59 |
| 35 | 2,31 | 5,64 | 2,58 |
| 40 | 3,51 | 8,55 | 3,93 |
| 45 | 5,05 | 12,30 | 5,67 |
| 50 | 7,01 | 17,07 | 7,88 |
| 55 | 9,39 | 22,93 | 10,58 |

Таблица 10

**Среднее время работы алгоритмов для задачи MIN-DISAGREE, с**

| $n$ | $p$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | 0,33 | | | 0,5 | | | 0,67 | | |
| | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** |
| 15 | 0,1 | 1,0 | 0,1 | 0,1 | 1,4 | 0,4 | 0,1 | 0,7 | 0,1 |
| 20 | 0,7 | 187,0 | 1,1 | 2,3 | 602,9 | 5,8 | 0,2 | 57,6 | 0,4 |
| 25 | 4,8 | 3787,5 | 9,4 | 22,6 | – | 50,2 | 0,8 | 2972,9 | 1,9 |
| 30 | 27,7 | – | 94,3 | 841,2 | – | 1227,3 | 3,2 | – | 11,2 |
| 35 | 440,5 | – | 971,8 | – | – | – | 6,4 | – | 19,9 |
| 40 | – | – | – | – | – | – | 19,9 | – | 264,4 |
| 45 | – | – | – | – | – | – | 58,6 | – | 866,8 |
| 50 | – | – | – | – | – | – | 204,1 | – | – |
| 55 | – | – | – | – | – | – | 519,3 | – | – |

Из табл. 9 видно, что для задачи MIN-DISAGREE больше всего памяти требуется модели (23)–(29). При $n = 50$ она занимает более 16,5 Мбайт, в то время как размер моделей (5)–(7) и (10)–(14) растёт медленнее и почти одинаково.

Таблица 11

**Границы доверительного интервала среднего времени работы алгоритмов для задачи MIN-DISAGREE, $p = 0{,}33$ и $p = 0{,}5$, с**

| $n$ | $p$ | | | | | |
|---|---|---|---|---|---|---|
| | 0,33 | | | 0,5 | | |
| | **TR** | **MOD** | **IN** | **TR** | **MOD** | **IN** |
| 15 | [0,1, 0,2] | [0,9, 1,2] | [0,1, 0,2] | [0,1, 0,2] | [1,3, 1,6] | [0,3, 0,5] |
| 20 | [0,7, 0,9] | [155,2, 231,4] | [0,9, 1,3] | [2,1, 2,5] | [525,3, 698,9] | [5,3, 6,4] |
| 25 | [4,3, 5,3] | [2303,9, 5508,2] | [8,3, 10,6] | [20,5, 25,4] | – | [46,2, 54,7] |
| 30 | [23,1, 34,8] | – | [79,9, 112,9] | [730,1, 1003,3] | – | [1112,6, 1461,2] |
| 35 | [328,1 636,4] | – | [795,4, 1479,3] | – | – | – |

При всех значениях параметра $p$ алгоритм **MOD** имеет наихудшее среднее время работы (табл. 10–12 и рис. 5). Так, при $p = 0{,}33$ и $n = 25$ среднее время его работы равно 3785,5 с, что в 789 раз больше, чем у алгоритма **TR**, и в 403 раза больше, чем у алгоритма **IN**. При всех значениях параметра $p$ среднее время работы алгоритма **TR** меньше, чем у алгоритма **IN**. Поскольку доверительные интервалы всех алгоритмов не пересекаются, то среднее время работы алгоритма **TR** статистически значимо наименьшее.

Таблица 12

**Границы доверительного интервала среднего времени работы алгоритмов для задачи MIN-DISAGREE, $p = 0{,}67$, с**

| $n$ | **TR** | **MOD** | **IN** |
|---|---|---|---|
| 15 | [0,0, 0,1] | [0,6, 0,7] | [0,0, 0,1] |
| 20 | [0,2, 0,3] | [45,6, 73,4] | [0,3, 0,6] |
| 25 | [0,6, 1,1] | [2457,2, 3697,9] | [1,4, 2,8] |
| 30 | [2,3, 4,8] | – | [7,1, 18,1] |
| 35 | [4,9, 8,9] | – | [13,8, 36,2] |
| 40 | [13,8, 31,7] | – | [125,2, 861,7] |
| 45 | [41,3, 87,5] | – | [315,8, 2355,5] |
| 50 | [125,7, 469,6] | – | – |
| 55 | [292,9, 1288,1] | – | – |

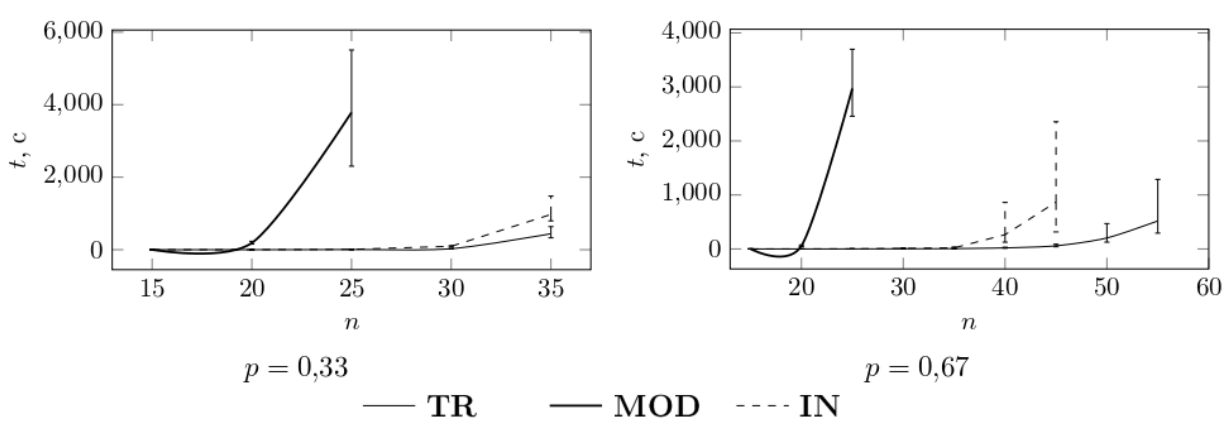


Рис. 5. Среднее время ($t$, с) работы алгоритмов для задачи MIN-DISAGREE

По результатам исследования можно сделать следующие выводы:

1) в задачах с ограничением числа кластеров наименьшее среднее время достижения оптимума принадлежит алгоритму **MOD**, при этом опорная модель ЦЛП требует небольшой объем памяти. Алгоритмы **TR** и **IN** требуют больше времени и памяти;
2) в задачах без ограничения числа кластеров наименьшее среднее время достижения оптимума принадлежит алгоритму **TR**, при этом опорная модель ЦЛП требует небольшой объем памяти. Алгоритм **IN** требует больше времени, а алгоритм **MOD** — больше времени и памяти;
3) все модели являются перспективными для дальнейшего теоретического исследования из-за разной структуры областей допустимых значений (например, для построения приближённых алгоритмов).

## Заключение

В работе рассматриваются задачи кластеризации вершин графа. Изучается подход к построению моделей ЦЛП для этих задач. Приведён обзор известных моделей ЦЛП, а также предложены новые подходы к их построению, один из которых позволяет значительно сократить количество неравенств для задачи с ограничением числа кластеров. Из результатов вычислительного эксперимента следует, что один из алгоритмов, опирающийся на новые модели, является наилучшим при поиске точных решений в задачах с ограничением числа кластеров. В то же время в варианте задачи без ограничений лучшие результаты принадлежат алгоритму, опирающемуся на известную модель. Все описанные модели представляют интерес для теоретического исследования в контексте построения приближённых алгоритмов.

## ЛИТЕРАТУРА

1. *Schaeffer S. E.* Graph clustering // Comput. Sci. Rev. 2005. V. 1. No. 1. P. 27–64.
2. *Bansal N., Blum A., and Chawla S.* Correlation clustering // Machine Learning. 2004. V. 56. P. 89–113.
3. *Shamir R., Sharan R., and Tsur D.* Cluster graph modification problems // Discrete Appl. Math. 2004. V. 144. No. 1–2. P. 173–182.
4. *Ben-Dor A., Shamir R., and Yakhimi Z.* Clustering gene expression patterns // J. Comput. Biol. 1999. V. 6. No. 3–4. P. 281–297.
5. *Solé P. and Zaslavsky T.* A coding approach to signed graphs // SIAM J. Discrete Math. 1994. No. 7. P. 544–553.
6. *Kriegel H. P., Kroger P., and Zimek A.* Clustering high-dimensional data // ACM Trans. Knowledge Discovery from Data. 2009. No. 3. P. 1–58.
7. *Balamurugan R., Natarajan A. M., and Premalatha K.* Stellar-mass black hole optimization for biclustering microarray gene expression data // Appl. Artif. Intell. 2015. V. 29. No. 4. P. 353–381.
8. *Křivánek M. and Morávek J.* NP-hard problems in hierarchical-tree clustering // Acta Informatica. 1986. V. 23. P. 311–323.
9. *Giotis I. and Guruswami V.* Correlation clustering with a fixed number of clusters // Theory of Computing. 2006. V. 2. No. 1. P. 249–266.
10. *Агеев А. А., Ильев В. П., Кононов А. В., Талевнин А. С.* Вычислительная сложность задачи аппроксимации графов // Дискретн. анализ и исслед. опер. Сер. 1. 2006. Т. 13. № 1. С. 3–11.

11. *Coleman T., Saunderson J., and Wirth A.* A local-search 2-approximation for 2-correlation-clustering // LNCS. 2008. V. 5193. P. 308–319.
12. *Ильев В. П., Ильева С. Д., Навроцкая А. А.* Приближенные алгоритмы для задач аппроксимации графов // Дискретн. анализ и исслед. опер. 2011. Т. 18. № 1. С. 41–60.
13. *Ильев В. П., Ильева С. Д., Моршинин А. В.* 2-Приближенные алгоритмы для двух задач кластеризации на графах // Дискретн. анализ и исслед. опер. 2020. Т. 27. № 3. С. 88–108.
14. *Ильев В. П., Ильева С. Д., Моршинин А. В.* Алгоритмы приближённого решения одной задачи кластеризации графа // Прикладная дискретная математика. 2019. № 45. С. 64–77.
15. *Charikar M., Guruswami V., and Wirth A.* Clustering with qualitative information // J. Comput. Syst. Sci. 2005. V. 71. No. 3. P. 360–383.
16. *Ailon N., Charikar M., and Newman A.* Aggregating inconsistent information: Ranking and clustering // J. ACM. 2008. V. 55. No. 5. P. 1–27.
17. *Chawla S., Makarychev K., Schramm T., and Yaroslavtsev G.* Near optimal LP algorithm for correlation clustering on complete and complete $k$-partite graphs // Proc. STOC'15. Portland, Oregon, USA, 2015. P. 219–228.
18. *Nishimura N., Ragde P., and Thilikos D. M.* On graph powers for leaf-labeled trees // J. Algorithms. 2002. V. 42. No. 1. P. 69–108.
19. *Ferneyhough S., Haas R., Hanson D., and MacGillivaray G.* Star forests, dominating sets and Ramsey-type problems // Discrete Math. 2002. V. 245. No. 1–3. P. 255–262.
20. *Wahid D. F. and Hassini E.* A literature review on correlation clustering: cross-disciplinary taxonomy with bibliometric analysis // Oper. Res. Forum. 2020. V. 3. No. 47. P. 1–42.
21. *Wiliams H. P. and Hong Y.* Representations of the all-different predicate of constraint satisfaction in integer programming // NFORMS J. Computing. 2001. V. 13. No. 2. P. 96–103.
22. *Bisschop J.* Aimms Optimization Modeling. `https://documentation.aimms.com/_downloads/AIMMS_modeling.pdf`. 2023.
23. *Harary F.* On the notion of balance of a signed graph // Michigan Math. J. 1953. No. 2. P. 143–146.
24. *Alon N. and Spencer J. H.* The Probabilistic Method. 4th ed. N.Y.: Wiley & Sons, 2016.
25. *Horowitz J. L.* Bootstrap methods in econometrics // Ann. Rev. Economics. 2019. V. 11. P. 193–224.
26. *Efron B.* Better bootstrap confidence intervals // J. ASA. 2014. V. 82. No. 397. P. 171–185.

## REFERENCES

1. *Schaeffer S. E.* Graph clustering. Comput. Sci. Rev., 2005, vol. 1, no. 1, pp. 27–64.
2. *Bansal N., Blum A., and Chawla S.* Correlation clustering. Machine Learning, 2004, vol. 56, pp. 89–113.
3. *Shamir R., Sharan R., and Tsur D.* Cluster graph modification problems. Discrete Appl. Math., 2004, vol. 144, no. 1–2, pp. 173–182.
4. *Ben-Dor A., Shamir R., and Yakhimi Z.* Clustering gene expression patterns. J. Comput. Biol., 1999, vol. 6, no. 3–4, pp. 281–297.
5. *Solé P. and Zaslavsky T.* A coding approach to signed graphs. SIAM J. Discrete Math., 1994, no. 7, pp. 544–553.
6. *Kriegel H. P., Kroger P., and Zimek A.* Clustering high-dimensional data. ACM Trans. Knowledge Discovery from Data, 2009, no. 3, pp. 1–58.
7. *Balamurugan R., Natarajan A. M., and Premalatha K.* Stellar-mass black hole optimization for biclustering microarray gene expression data. Appl. Artif. Intell., 2015, vol. 29, no. 4, pp. 353–381.

8. *Křivánek M. and Morávek J.* NP-hard problems in hierarchical-tree clustering. Acta Informatica, 1986, vol. 23, pp. 311–323.
9. *Giotis I. and Guruswami V.* Correlation clustering with a fixed number of clusters. Theory of Computing, 2006, vol. 2, no. 1, pp. 249–266.
10. *Ageev A. A., Il'ev V. P., Kononov A. V., and Talevnin A. S.* Computational complexity of the graph approximation problem. J. Appl. Industr. Math., 2007, vol. 1, no. 1, pp. 1–8.
11. *Coleman T., Saunderson J., and Wirth A.* A local-search 2-approximation for 2-correlation-clustering. LNCS, 2008, vol. 5193, pp. 308–319.
12. *Il'ev V. P., Il'eva S. D., and Navrotskaya A. A.* Approximation algorithms for graph approximation problems. J. Appl. Industr. Math., 2011, vol. 5, no. 4, pp. 569–581.
13. *Il'ev V. P., Il'eva S. D., and Morshinin A. V.* 2-Approximation algorithms for two graph clustering problems. J. Appl. Industr. Math., 2020, vol. 14, no. 3, pp. 490–502.
14. *Il'ev V. P., Il'eva S. D., and Morshinin A. V.* Algoritmy priblizhennogo resheniya odnoy zadachi klasterizatsii grafa [Approximate algorithms for graph clustering problem]. Prikladnaya Diskretnaya Matematika, 2019, no. 45, pp. 64–77. (in Russian)
15. *Charikar M., Guruswami V., and Wirth A.* Clustering with qualitative information. J. Comput. Syst. Sci., 2005, vol. 71, no. 3, pp. 360–383.
16. *Ailon N., Charikar M., and Newman A.* Aggregating inconsistent information: Ranking and clustering. J. ACM, 2008, vol. 55, no. 5, pp. 1–27.
17. *Chawla S., Makarychev K., Schramm T., and Yaroslavtsev G.* Near optimal LP algorithm for correlation clustering on complete and complete $k$-partite graphs. Proc. STOC'15, Portland, Oregon, USA, 2015, pp. 219–228.
18. *Nishimura N., Ragde P., and Thilikos D. M.* On graph powers for leaf-labeled trees. J. Algorithms, 2002, vol. 42, no. 1, pp. 69–108.
19. *Ferneyhough S., Haas R., Hanson D., and MacGillivaray G.* Star forests, dominating sets and Ramsey-type problems. Discrete Math., 2002, vol. 245, no. 1–3, pp. 255–262.
20. *Wahid D. F. and Hassini E.* A literature review on correlation clustering: cross-disciplinary taxonomy with bibliometric analysis. Oper. Res. Forum, 2020, vol. 3, no. 47, pp. 1–42.
21. *Wiliams H. P. and Hong Y.* Representations of the all-different predicate of constraint satisfaction in integer programming. NFORMS J. Computing, 2001, vol. 13, no. 2, pp. 96–103.
22. *Bisschop J.* Aimms Optimization Modeling. `https://documentation.aimms.com/_downloads/AIMMS_modeling.pdf`, 2023.
23. *Harary F.* On the notion of balance of a signed graph. Michigan Math. J., 1953, no. 2, pp. 143–146.
24. *Alon N. and Spencer J. H.* The Probabilistic Method, 4th ed. N.Y., Wiley & Sons, 2016.
25. *Horowitz J. L.* Bootstrap methods in econometrics. Ann. Rev. Economics, 2019, vol. 11, pp. 193–224.
26. *Efron B.* Better bootstrap confidence intervals. J. ASA, 2014, vol. 82, no. 397, pp. 171–185.