Алгоритмы локального поиска в задаче исследования инвариантов графа

А. С. Белозеров, Б. Ф. Мельников, Н. П. Чурикова

***Аннотация* — Нерешенная проблема изоморфизма графов напрямую связана с одной из семи т.н. проблем тысячелетия: с проблемой равенства классов P и NP из теории алгоритмов. Разрешение связанных между собой проблем обусловлено исходом упорного поиска алгоритма сертификата для проверки двух графов на изоморфизм.**

Количественную характеристику структуры графа называют инвариантом графа. При этом инвариант называют полным, если равенство его значений для двух разных графов означает изоморфность рассматриваемых графов. Известные на данный момент полные инварианты (например, т.н. макси-код) являются трудновычислимыми и не позволяют эффективно решать задачу определения изоморфизма графов.

На практике применима более простая процедура, в основе которой лежит построение канонического кода графа за квазиполиномиальное время. Перебор разбиений при нахождении канонической перестановки существенно сокращается при использовании группы автоморфизмов графа.

Алгоритмизация задачи по генерации графов на основе модификации несколькими видами их матрицы смежности и определении некоторых инвариантов графа по критерию наибольшей информативности позволила провести расчетные исследования. Анализ результатов экспериментальных расчетов выявил информативность следующих инвариантов для дифференциации графов: индекс Винера, определитель матрицы смежности и с меньшей степенью информативности диаметр. Статистически выявлен низкий уровень информативности таких инвариантов графа, как вектор степеней второго порядка, число компонент связности, радиус, индекс Рандича. Видится актуальной статистическая задача по исследованию парных корреляций для этих инвариантов.

***Ключевые слова* — связный неориентированный граф, инвариант графа, эвристический алгоритм, индекс Рандича, изоморфизм графов.**

I. Введение

Нерешенная проблема изоморфизма графов остается основополагающей теоретической задачей для разрешения одной из так называемых проблем тысячелетия – проблемы равенства классов P и NP [1].

А. С. Белозеров, Федеральное государственное автономное образовательное учреждение высшего образования «Московский физико-технический институт (государственный университет)», Москва, РФ (e-mail: keizoff@icloud.com).

Б. Ф. Мельников, Московский государственный университет — Пекинский политехнический институт, Шэньчжэнь, Провинция Гуандун, КНР (e-mail: bf-melnikov@yandex.ru).

Н. П. Чурикова, Федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С. П. Королева», Самара, РФ (e-mail: claorisel@gmail.com).

Задача проверки изоморфизма графов принадлежит к классу задач, для которых пока неизвестно, являются ли они полиномиально разрешимыми или нет, т. е. пока не построен полиномиальный алгоритм проверки изоморфизма любых графов. В 2015 году Л. Бабай анонсировал алгоритм, позволяющий решить задачу изоморфизма за квазиполиномиальное время $\exp(\log(n)^{O(1)})$, где n – число вершин графа) [2]; в 2017 году этот алгоритм был уточнен [3].

Непосредственная проверка на изоморфизм двух n-вершинных графов заключается в рассмотрении всех $n!$ перестановок вершин и установлении совместимости рёбер графов хотя бы при одной перестановке. Такой подход, даже при небольшом количестве вершин, требует огромных вычислительных ресурсов.

Изоморфизм можно рассматривать как перенумерацию вершин графа, поэтому любая количественная характеристика структуры графа остается неизменной. Такие характеристики называются инвариантами графа. Среди множества инвариантов графа выделяются полные инварианты: инвариант называют полным, если равенство его значений для двух разных графов означает изоморфность данных графов [4]. Известные на данный момент полные инварианты (например, макси-код) являются трудновычислимыми [5] и не позволяют эффективно решать задачу определения изоморфизма графов.

На практике применяются более простые процедуры, от которых ожидается хорошая работа в большинстве случаев. Например, используются специальные эвристики для доказательства, что некоторые два графа неизоморфны [6].

Существует подход к исследованию графов на изоморфизм, в основе которого лежит построение канонического кода графа, не зависящего от порядка нумерации вершин [7, 8]. На основе такого подхода в 1981 году Б. МакКей разработал программу nasty [7], а затем программу traces [8]. Два графа изоморфны тогда и только тогда, когда совпадают их канонические коды, которым соответствует каноническая нумерация вершин. Для решения этой задачи в пакете nauty используется представление множества всех вершин графа в виде разбиения – упорядоченного набора непересекающихся ячеек (подмножеств вершин).

Перебору всех перестановок вершин соответствует перебор и уточнение разбиений, в результате которых появляется *каноническая нумерация вершин*. Уточнение разбиений сопоставляется обходу по дереву разбиений (дереву поиска). Перебор разбиений при нахождении канонической перестановки существенно сокращается

при использовании группы автоморфизмов графа.

Составной частью nauty является генерация автоморфизмов графа, в результате действия которых появляются орбиты вершин – подмножества вершин, в которые переходят заданные вершины. С помощью nauty можно найти каноническую нумерацию, построить автоморфизмы в неориентированных и ориентированных графах (а также в раскрашенных графах) и провести исследование графов на изоморфизм [9].

К существующим современным программам авторами статьи реализован генератор графов на основе модификации несколькими видами их матрицы смежности. Алгоритмизация задачи по количественному определению информативности заданного набора инвариатов сгенерированных графов позволила провести расчетные исследования по критерию наибольшей информативности при исследование графов на изоморфизм.

Анализ результатов экспериментальных расчетов выявил информативность таких инвариантов графов как индекс Винера, определитель матрицы смежности и диаметр, но с меньшей степенью информативности последнего. В качестве исходных графов для генерации выборки графов мы рассмотрели как планарные, так и не планарные графы.

II. Задача изоморфизма графов в теории сложности

Теория сложности занимается изучением бесконечных семейств задач, в каждом из которых любая задача может быть решена с помощью одного и того же алгоритма. Значительное упрощение алгоритма или многократное увеличение его быстродействия эвристическими способами (догадкой или искусным приемом) возможно для алгоритмически решаемых по своей природе задач. Существуют определенные классы проблем, решать которые с помощью алгоритмов несоизмеримо труднее. Их решают либо с помощью очень медленных алгоритмов, либо с помощью алгоритмов, требующих чрезмерно больших ресурсов для хранения информации [10].

В теории сложности можно рассмотреть три класса задач: P, NP и NPC (класс NP-полных задач) [11]. Класс P состоит из задач, разрешимых в течение полиномиального времени $O(n^k)$, где k – некоторая константа, а n – размер входных данных задачи. Класс NP состоит из задач, которые поддаются проверке в течение полиномиального времени. Имеется в виду, что если мы каким-то образом получаем «сертификат» решения, то в течение времени, полиномиальным образом зависящего от размера входных данных задачи, можно проверить корректность такого решения.

Пример NP-задачи: нахождение «гамильтонова цикла» на графе. В настоящей работе термин «граф» означает связный неориентированный граф без петель и кратных ребер. Гамильтонов цикл – это замкнутый маршрут (петля), состоящий только из сторон графа и проходящий не более одного раза через любую из вершин. Пример графа с изображенным на нем гамильтоновым циклом показан на рисунке 1. В задаче о гамильтоновом цикле с заданным ориентированным графом $G = (V, E)$ сертификат имеет вид последовательности $(v_1, v_2, \ldots, v_{|V|})$ из $|V|$ вершин. В течение полиномиального времени легко проверить, что

$$(v_i, v_{(i+1)}) \in E,$$

при $i = 1, 2, \ldots, |V| - 1$ и что

$$(v_{|V|}, v_1) \in E.$$

Если бы удалось отыскать алгоритм для решения задачи нахождения гамильтонова цикла за «полиномиальное» время, то это будет означать, что все NP-задачи будут лежать в P.

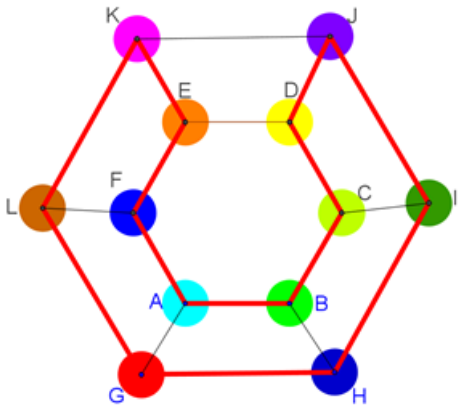


Рис. 1. Гамильтонов цикл на планарном графе

Задача проверки изоморфизма графов среди прочих NP-полных задач представляют особый интерес. Во-первых, потому, что есть псевдоэффективный алгоритм ее решения (аналог метода ветвей и границ), в то время как для других NP-полных задач пока неизвестно о существовании эффективных алгоритмов их решения. Во-вторых, если эффективный алгоритм существует хотя бы для одной из NP-полных задач, то его можно сформулировать и для всех остальных. Любая задача класса P принадлежит классу NP, поскольку принадлежность задачи классу P означает, что ее решение можно получить в течение полиномиального времени, даже не располагая сертификатом. Остается открытым вопрос, является ли P строгим подмножеством NP [11].

Большинство ученых, занимающихся теорией вычислительных систем, считают NP-полные задачи очень трудноразрешимыми, потому что при огромном разнообразии изучавшихся до настоящего времени NP-полных задач ни для одной из них пока так и не найдено решение в виде алгоритма с полиномиальным временем работы. Таким образом, было бы крайне удивительно, если бы все они оказались разрешимыми в течение полиномиального времени [5].

Другая задача, также являющаяся NP-полной – «задача коммивояжера», которая во многом похожа на гамильтонов цикл, но ставится цель отыскать гамильтонов цикл с минимальной «длиной» пути, проделанного коммивояжером. Полиномиальное время решения, достигнутое в «задаче коммивояжера», привело бы к возможности решать все NP-задачи за «полиномиальное» время. Если эта задача окажется решаемой за «полиномиальное» время, то, шифры могут быть взломаны при помощи мощных современных компьютеров [12].

Эксперты, как правило, полагают, что, используя устройство, работающее по принципу машины Тьюринга, невозможно за «полиномиальное» время решить NP-полную задачу; и что, следовательно, P и NP – неэкви-

валентны, хотя пока этот факт никто не смог доказать. И это остается наиболее важной и на сегодняшний день нерешенной задачей теории сложности [13].

III. Алгоритм сертификата для проверки двух N-вершинных графов на изоморфизм

В общем случае задача проверки двух графов на изоморфизм имеет вид: даны графы G_1 и G_2 с множеством вершин $V(G_1)$, $V(G_2)$ и множеством ребер $E(G_1)$ и $E(G_2)$. $|V(G_1)| = |V(G_2)| = n$, $|E(G_1)| = |E(G_2)|$. Необходимо определить, существует ли биективное отображение $\varphi: V_A \to V_B$ такое, что

$$(i,j) \in E(G_1) \Leftrightarrow (\varphi(i), \varphi(j)) \in E(G_2).$$

В настоящее время все еще не существует алгоритма, позволяющего определить изоморфность произвольных графов за полиномиальное время [14]. Для доказательства того, что два графа неизоморфны, используются различные эвристики, например, используют различные простые инварианты, и, как только обнаруживаются два различных значения одного и того же инварианта, приходят к заключению, что графы неизоморфны.

В [15] был предложен один из вариантов проверки двух графов на неизоморфность – а именно, эвристический алгоритм определения конкретной последовательности проверки инвариантов. В настоящей работе рассматриваются следующие инварианты графов [16]:

A. Индекс Рандича

$$\sum_{(v_i,v_j)\in V} \frac{1}{\sqrt{d(v_i)d(v_j)}}$$

где v_i и v_j – вершины, образующие ребро, $d(v_k)$ – степень вершины v_k.

B. Вектор степеней второго порядка – вектор, каждый элемент, которого представляет собой список степеней вершин, смежных с данной.

В ходе исследования нами была создана база данных графов, содержащая все простые связные графы на 8, 9 и 10 вершинах, количество таких графов 11117, 261080, 11716571 соответственно. Для хранения графов используется формат graph6 – тестовый формат для хранения неориентированных графов. Формат graph6 (g6) поддерживается программой nauty [17]. Для генерации базы графов использовалась программа nauty, с ее помощью были созданы файлы, содержащие все простые связные графы на заданное количество вершин в формате g6.

Программа для вычислений реализована на языке программирования Python 3.6, для хранения полученных результатов используется система управления базами данных MongoDB. Данная СУБД использует для хранения данных формат BSON, являющийся бинарным аналогом JSON, что позволяет эффективно описывать сложные структуры данных, быстрее выполнять операции поиска и обработки.

Функция вычисления инвариантов для графа в формате g6 и записи в базу данных на языке Python выглядит следующий образом:

```
defcomputing_gr(graph):
    g = nx.parse_graph6(graph)
    # количество ребер
edges = len(nx.edges(g))
    # значение инварианта индекс Рандича
    # число с плавающей точкой, с точностью до 6
randic = invariants_tools.get_randic_index(g)
# вектор степеней второго порядка
    # отсортированный список
sec_vec = invariants_tools.get_sec_vec(g)
    # запись в базу данных
mongo_tools.insert_base_item(posts, graph,
str(edges), str(randic), str(sec_vec))
```

Так как вычисления для такого количества графов весьма затратны по времени и вычислительным ресурсам, программа была написана с использованием парадигмы параллельного программирования. Вычисления инвариантов для графов производились одновременно в 4 потока, полностью используя вычислительные ресурсы машины. Для каждого графа из базы были вычислены 2 вышеописанных инварианта. Обработка данных в 4 потока:

```
defcomputing_gr(graph):
pool = Pool(4)
pool.map(computing_gr, read_input(g6f))
pool.close()
pool.join()
```

В итоге для каждого графа в базе хранится запись следующего вида:

```
{ "_id" : ObjectId("5be6be0ebc54f4673d5900c0"),
"sec_vec" : "[[2, 1, 1, 1, 1], [5, 2], [2, 1],
[5], [5], [5], [5], [2]]", "graph6" : "G??EDw",
"randic" : "3.3121889292032174", "date" : ISO-
Date("2018-11-10T11:16:30.618Z"), "nodes" : "7"
}
```

После создания базы данных проведена работа по поиску пар графов, дифференцируемых индексом Рандича или вектором степеней второго порядка. Функция поиска данных графов:

```
#на вход функция получает все возможные уникаль-
ные варианты пар графов
deffind_base_result(pair):
 …
if (sec_vec_0 == sec_vec_1):
ifmath.isclose(randic_0, randic_1,
abs_tol=0.000001) is False:
if g6_0 != g6_1:
res_srt = "A: {0} B: {1} \n".format(g6_0, g6_1)
fl_randic_base.write(res_srt)
else:
if math.isclose(randic_0, randic_1,
abs_tol=0.0001) is True:
if g6_0 != g6_1:
res_srt = "A: {0} B: {1} \n".format(g6_0, g6_1)
fl_sec_vec_base.write(res_srt)
```

Рассмотрим работу программы на примере графов с 8 вершинами.

АЛГОРИТМ 1. ОПИСАНИЕ АЛГОРИТМА ВОССТАНОВЛЕНИЯ СЛУЧАЙНОГО ГРАФА ПО ЗАДАННОМУ ВЕКТОРУ СТЕПЕНЕЙ.

Вход: список всех графов с 8 вершинами.

1. По очереди выбираются графы с заданным количеством ребер (для графов с 8 вершинами – от 7 до 28 ребер).
2. Для набора с заданным количество ребер строится набор уникальных пар графов.
3. Для каждой пары из набора вызывается функция поиска заданных графов, find_base_result описанная выше.

Выход: все пары графов проверены.

(Конец описания алгоритма)

В результате работы программы получили набор файлов, по два файла на каждое количество ребер в графе. Для графов с 8 вершинами это будет 42 файла. Имя файла включает в себя информацию о количестве ребер, количестве вершин и название инварианта. То есть, файл с названием 8_result_nodes_sec_vec_14.txt содержит в себе такие пары графов, для которых вектор степеней второго порядка отличается, а индекс Рандича совпадает.

В результате вычислений для простых связных графов с 8, 9 и 10 вершинами были найдены примеры графов [16] для которых совпадает индекс Рандича, но не совпадает вектор степеней второго порядка. Обратных примеров, когда совпадает вектор степеней второго порядка, но не совпадает индекс Рандича, найдено не было.

IV. ГЕНЕРАТОР ГРАФОВ

Генератор графов реализован нами как графическое десктопное приложение на платформе WPF (система для построения клиентских приложений Windows) инструментальными средствами API-интерфейса с подключением к базе данных от провайдера MS SQL Server. Логика программы реализована на языке программирования C# под случайную генерацию матрицы смежности графа. За отрисовку элементов графики отвечали такие части ОС Windows, как DirectX. Рендеринг графики в WPF ложится на графический процессор на видеокарте, что позволяет воспользоваться аппаратным ускорением графики.

Идея способа генерации графов принадлежит руководителю авторского состава. Полученные и описанные в статье результаты являются косвенным подтверждением того, что этот способ вовсе неплох. Предложенным способом количество графов в БД довели до 25. Для каждого графа провели вычисление его следующих инвариантов: вектор степеней, определитель матрицы смежности, число компонент связности, диаметр, радиус, индекс Рандича, индекс Винера [18].

Первая часть генерации графов при формировании набора графов в базе данных выполнена на основе модификации матрицы смежности для каждого нового генерируемого графа по правилу перехода

$$\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} \to \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}.$$

Так как матрица смежности симметрична, то переходы были выполнены как в верхнетреугольной, так и в нижнетреугольной части матрицы смежности графа одновременно. Для сгенерированных таким образом графов выполнена вторая часть модификации матрицы смежности.

Случайно выбираем 6 вершин, пусть это будут вершины D, F, H, I, J, L, такие, что выполняется

$$\begin{array}{c} \\ \mathrm{D}\\ \mathrm{F}\\ \mathrm{H}\end{array}\begin{array}{c}\begin{array}{ccc}\mathrm{I} & \mathrm{J} & \mathrm{L}\end{array}\\ \begin{pmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 1 & 0 & 0\end{pmatrix}\end{array}$$

и производим замену на

$$\begin{array}{c} \\ \mathrm{D}\\ \mathrm{F}\\ \mathrm{H}\end{array}\begin{array}{c}\begin{array}{ccc}\mathrm{I} & \mathrm{J} & \mathrm{L}\end{array}\\ \begin{pmatrix}0 & 0 & 1\\ 1 & 0 & 0\\ 0 & 1 & 0\end{pmatrix}\end{array}.$$

Модификации матриц выполнены как в верхнетреугольной ее части, так и в нижнетреугольной части матрицы смежности графа одновременно.

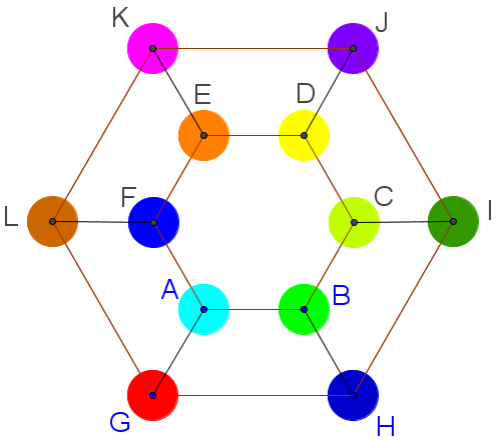


Рис. 2. Исходный симметричный планарный граф

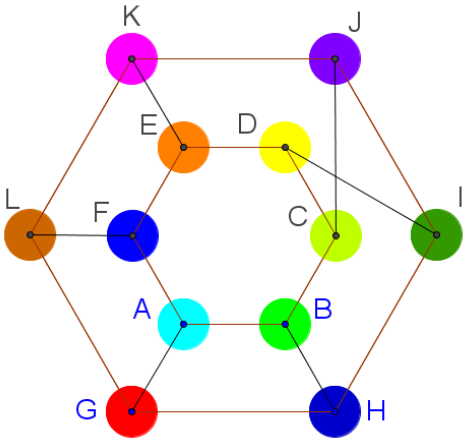


Рис. 3. Граф после модификации симметричного планарного графа первым способом

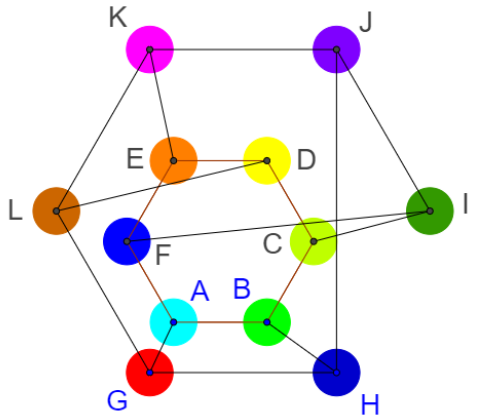


Рис. 4. Граф после модификации симметричного планарного графа вторым способом

Код процедуры, которая выполняет добавление данных в БД:

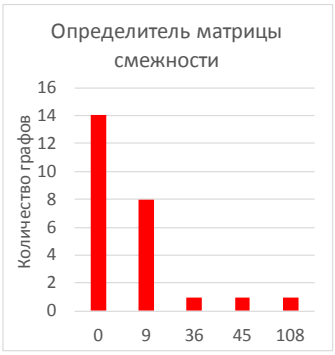
```
CREATE PROCEDURE [dbo].[sp_InsertGraph]
    @AdjacencyMatrix text,
    @NumberOfPoints int,
    @NumberOfEdges int,
    @VectorOfPowers text,
    @IdentifierAdjacencyMatrix float,
    @NumberOfConnectivityComponents int,
    @Diameter int,
    @Radius int,
    @RandicIndex float,
    @WienerIndex float,
    @IdGraph int out
AS
    INSERT INTO Graphs (AdjacencyMatrix, NumberOfPoints, NumberOfEdges, VectorOfPowers, IdentifierAdjacencyMatrix, NumberOfConnectivityComponents, Diameter, Radius, RandicIndex, WienerIndex)
    VALUES (@AdjacencyMatrix, @NumberOfPoints, @NumberOfEdges, @VectorOfPowers, @IdentifierAdjacencyMatrix, @NumberOfConnectivityComponents, @Diameter, @Radius, @RandicIndex, @WienerIndex)

    SET @IdGraph =SCOPE_IDENTITY()
GO
```

При подключении из кода C# процедура упрощает работу, возвращая IdGraph добавленной записи при добавлении нового объекта в базу данных.

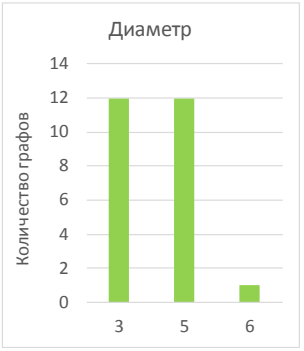
V. Статистическая обработка в поисках дифференцированности графов

Исследуя несколько инвариантов графа по критерию наибольшей информативности, то есть, какой из них «более дифференцирующий» или, другими словами, дающий ответ о неизоморфности графов, если те действительно неизоморфны – с большей вероятностью, подошли к гипотезе об отсутствии такого [19, 20]. В этой ситуации видятся актуальными статистическая обработка данных во множестве сгенерированных графов по количественному определению более или менее информативного инварианта. Из всего перечня исследованных инвариантов разброс значений наблюдали для трех, а именно, для определителя матрицы смежности, для диаметра и для индекса Винера.

Плотности распределения графов по значениям своих инвариантов представлены на диаграммах ниже (см. рис. 5). Заметим, что индекс Винера наиболее информативен и дифференцируем, в отличие от всех остальных. В меньшей степени информативность у определителя матрицы смежности, еще меньше у такого инварианта, как диаметр и далее в порядке убывания у всех остальных из рассмотренных нами.



а

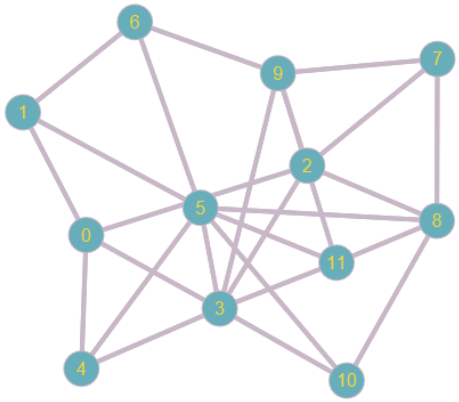


б

в

Рис. 5. Плотности распределения графов по значениям своих инвариантов: по определителю матрицы смежности (*а*), по диаметру (*б*), по индексу Винера (*в*)

Как было замечено авторами статьи, статистика планарного графа несколько отличается от статистических характеристик непланарного графа, принятого за исходный для генерации набора графов. Так, например, возьмем за исходный граф для генератора графов произвольный непланарный граф с тем же количеством вершин согласно рисунку 6 (*а*). Плотности распределения графов, сгенерированных двумя вышеописанными способами, по значениям своих инвариантов представлены на диаграммах согласно рисункам 6 (*б*) и 6 (*в*). Наблюдаем, что индекс Винера менее информативен и дифференцируем в сравнении с определителем матрицы смежности. Все остальные из рассмотренных нами инвариантов оказались не чувствительными к произведенным модификациям матрицы смежности при определении изоморфности графов из полученного набора.



а

б

в

Рис. 6 – Плотности распределения графов по значениям своих инвариантов: по определителю матрицы смежности (*а*), по диаметру (*б*), по индексу Винера (*в*)

VI. Заключение

Непосредственное тестирование двух библиотек nauty и Traces на связных графах из 8, 9 и 10 вершин для проверки их на изоморфизм показало преимущество и эффективность процедуры, в основе которой лежит построение канонического кода графа. Для рассмотренных относительно «маленьких» графов каноническая разметка считается довольно быстро, но плохо подходит для общего решения проблемы изоморфизма на коллекции графов.

По этой причине авторы работы предложили свой способ генерации графов, который подтвердил свою состоятельность для генерации коллекции графов на 12 вершинах. Алгоритмизация задачи по генерации графов на основе модификации несколькими видами их матрицы смежности и определении некоторых инвариантов графа по критерию наибольшей информативности позволила провести расчетные исследования. Анализ результатов экспериментальных расчетов выявил качественное сравнение информативности между инвариантами:

- индекс Винера наиболее информативен и дифференцируем, в отличие от остальных рассмотренных инвариантов;
- средняя степень информативности у определителя матрицы смежности и диаметра;
- низкий уровень информативности у вектора степеней второго порядка, у числа компонент связности, у радиуса, у индекса Рандича;
- вектор степеней второго порядка лучше индекса Рандича.

Авторы обратили внимание на актуальность статистических задач по количественному определению более и менее информативного инварианта и по исследованию парных корреляций для таких изученных инвариантов, как вектор степеней, определитель матрица смежности, число компонент связности, диаметр, радиус, индекс Рандича, индекс Винера.

Библиография

[1] *Millennium Problems.* Clay Mathematics Institute, [URL]: https://claymath.org/millennium-problems, Дата обращения: 14.08.19.

[2] Babai L. *Graph isomorphism in quasipolynomial time.* [URL]: https://arxiv.org/pdf/1512.03547.pdf, Дата обращения: 14.08.19.

[3] *Graph Isomorphism in Quasipolynomial Time.* Wikimedia Foundation, Inc, [URL]: https://ru.wikipedia.org/wiki/Бабаи,_Ласло, Дата обращения: 14.08.19.

[4] Харари Ф. *Теория графов.* М.: Мир, 1973.

[5] Громкович Ю. *Теоретическая информатика.* Пер. с нем. / Под ред. Б. Ф. Мельникова, 3-е изд. СПб.: БХВ-Петербург, 2010.

[6] Мельников Б.Ф., Сайфуллина Е.Ф. *Применение мультиэвристического подхода для случайной генерации графов с заданным вектором степеней*. Известия высших учебных заведений, Поволжский регион, Физико-математические науки, 2013, №. 3(27), С.70-83.

[7] McKay B.D. *Practical graph isomorphism.* Congressus Numerantium, 1981, V. 30, P. 45-87.

[8] McKay B.D., Piperno A. *Practical graph isomorphism.* J. Symbolic Computation, 2014, V. 60, P. 94-112.

[9] Володичева М.И., Леора С.Н. *Исследование изоморфизма графов с помощью жордановых форм матриц смежности.* ПДМ, 2018, № 40, С.87-99.

[10] Пенроуз Р. *Новый ум короля: О компьютерах, мышлении и законах физики.* Пер. с англ. / Общ. ред. В. О. Малышенко. М.: Едиториал УРСС, 2003, 384 с.

[11] Кормен Т.Х., Лейзерсон Ч.И., Ривест. Р.Л., Штайн К. *Алгоритмы: построение и анализ.* 2-е изд., Пер. с англ., М.: Издательский дом «Вильямс», 2005.

[12] Гарднер М. *От мозаик Пенроуза к надежным шифрам.* Пер. с англ. М.: Мир, 1993.

[13] Gera R., Hedetniemi S., Larson C.E. *Graph Theory. Favorite Conjectures and Open Problems – 1.* Springer International Publishing Switzerland, 2016.

[14] McKay B., Piperno A. *Руководство пользователя nauty.* [URL]: http://users.cecs.anu.edu.au/~bdm/nauty, Дата обращения: 14.08.19.

[15] Мельников Б.Ф., Сайфуллина Е.Ф. *Генерация графов с заданным вектором степеней второго порядка и задача проверки изоморфизма*. Стохастическая оптимизация в информатике, 2014, № 2, С. 24-36.

[16] Чурикова Н.П. *О дифференциации графов на основе быстро вычисляемых инвариантов.* Материалы XII Белорусской математической конференции, 2016, Ч. 4, С. 67-69.

[17] Babai L., Erdos P., Selkow S.M. *Random graph isomorphism.* SIAM Journal on Computing, 1980, Vol. 9(3), P. 628-635.

[18] Мельников Б.Ф., Мельникова Е.А. *Кластеризация ситуаций в алгоритмах реального времени для задач дискретной оптимизации*. Системы управления и информационные технологии, 2007, Т. 28, № 2, С. 16-20.

[19] Мельников Б.Ф., Сайфуллина Е.Ф. *Применение мультиэвристического подхода для случайной генерации графа с заданным вектором степеней.* Известия высших учебных заведений, Поволжский регион, Физико-математические науки, 2013, № 3 (27), С. 70-83.

[20] Мельников Б.Ф., Сайфуллина Е.Ф., Терентьева Ю.Ю., Чурикова Н.П. *Применение алгоритмов генерации случайных графов для исследования надёжности сетей связи*. Информатизация и связь, 2018, № 1, С. 71-80.

Algorithms for local search in the problem of studying invariants of a graph

A. S. Belozerov, B. F. Melnikov, N. P. Churikova

***Abstract* — The unsolved problem of isomorphism of graphs is directly related to one of seven so-called millenium problems: with the problem of equality of classes P and NP from the theory of algorithms. The resolution of related problems is due to the outcome of a persistent search for the certificate algorithm for checking two graphs for isomorphism.**

The quantitative characteristic of the structure of a graph is called a graph invariant. Moreover, an invariant is called complete if the equality of its values for two different graphs means the isomorphism of the graphs under consideration. Currently known complete invariants (for example, the so-called maxi code) are difficult to calculate and do not effectively solve the problem of determining graph isomorphism.

In practice, a simpler procedure is applicable, based on the construction of the canonical code of a graph in quasi-polynomial time. Enumeration of partitions when finding the canonical permutation is significantly reduced when using the automorphism group of the graph.

Algorithmization of the problem of graph generation on the basis of modifying several types of their adjacency matrix and determining some graph invariants according to the criterion of the greatest informativeness made it possible to carry out computational studies. An analysis of the results of experimental calculations revealed the information content of the following invariants for graph differentiation: the Wiener index, the adjacency matrix determinant, and the diameter with a lower degree of information content. Statistically, a low level of informativeness of such invariants of the graph, as a vector of second-order degrees, the number of connected components, radius, Randic index, is revealed. It seems to be a relevant statistical task for the study of pair correlations for these invariants.

***Keywords* — connected undirected graph, graph invariant, heuristic algorithm, Randic index, graph isomorphism.**

REFERENCES

[1] *Millennium Problems*. Clay Mathematics Institute, [URL]: https://claymath.org/millennium-problems, Date of access: 08.14.19.

[2] Babai L. *Graph isomorphism in quasipolynomial time*. [URL]: https://arxiv.org/pdf/1512.03547.pdf, Date of access: 08.14.19.

[3] *Graph Isomorphism in Quasipolynomial Time*. Wikimedia Foundation, Inc, [URL]: https://en.wikipedia.org/wiki/Babai_Laslo, Date of access: 08.14.19.

[4] Kharari F. *Graph theory*. M.: Mir, 1973.

[5] Gromkovich Yu. *Theoretical informatics*. Translated from German by Melnikov B.F., 3rd ed. SPb .: BHV-Petersburg, 2010.

[6] Melnikov B.F., Sayfullina E.F. *Application of a multi-heuristic approach for random generation of graphs with a given vector of degrees*. University proceedings, Volga region, Physics and mathematics, 2013, no. 3 (27), pp. 70-83.

[7] McKay B.D. *Practical graph isomorphism.* Congressus Numerantium, 1981, V. 30, P. 45-87.

[8] McKay B.D., Piperno A. *Practical graph isomorphism*. J. Symbolic Computation, 2014, V. 60, P. 94-112.

[9] Volodicheva M.I., Leora S.N. *The study of graph isomorphism using Jordan forms of adjacency matrices*. PDM, 2018, No. 40, pp. 87-99.

[10] Penrose R. *The King's New Mind: On Computers, Thinking, and the Laws of Physics*. Translated from English by Malyshenko V.O., M.: URSS editorial, 2003, 384 p.

[11] Cormen, T.H., Leiserson, C.I., Rivest. RL, Stein K. *Algorithms: construction and analysis*. 2nd ed., Translated from English., M.: Williams Publishing House, 2005.

[12] Gardner M. *From Penrose Mosaics to Strong Ciphers*. Per. from English M .: Mir, 1993.

[13] Gera R., Hedetniemi S., Larson C.E. Graph Theory. *Favorite Conjectures and Open Problems - 1*. Springer International Publishing Switzerland, 2016.

[14] McKay B., Piperno A. *Nauty User Guide*. [URL]: http://users.cecs.anu.edu.au/~bdm/nauty; Access date: 08.14.19.

[15] Melnikov B.F., Sayfullina E.F. Generation of graphs with a given second-order degree vector and the problem of checking isomorphism. Stochastic Optimization in Computer Science, 2014, No. 2, C. 24-36.

[16] Churikova N.P. *On differentiation of graphs based on quickly calculated invariants*. Materials of the XII Belarusian Mathematical Conference, 2016, Part 4, S. 67-69.

[17] Babai L., Erdos P., Selkow S.M. *Random graph isomorphism*. SIAM Journal on Computing, 1980, Vol. 9 (3), P. 628-635.

[18] Melnikov B.F., Melnikova E.A. *Clustering situations in real-time algorithms for discrete optimization problems*. Management Systems and Information Technology, 2007, T. 28, No. 2, S. 16-20.

[19] Melnikov B.F., Sayfullina E.F. *Application of a multi-heuristic approach for random graph generation with a given degree vector*. University proceedings, Volga region, Physics and mathematics, 2013, No. 3 (27), pp. 70-83.

[20] Melnikov B.F., Sayfullina E.F., Terentyeva Yu.Yu., Churikova N.P. *The use of random graph generation algorithms to study the reliability of communication networks*. Informatization and Communication, 2018, No. 1, pp. 71-80.

Artem Sergeyevich BELOZEROV – a first year student of the Moscow Institute of Physics and Technology (https://mipt.ru/), email: keizoff@icloud.com.

Boris Feliksovich MELNIKOV – Professor of Moscow State University – MSU-PPI in Shenzhen (http://szmsubit.ru/), email: bf-melnikov@yandex.ru, mathnet.ru: personid=27967, elibrary.ru: authorid=15715, scopus.com: authorId=55954040300.

Nadezhda Petrovna CHURIKOVA – post-graduate student of the Federal state Autonomous educational institution of higher education "Samara national research University named after academician S. P. Korolev" (https://ssau.ru/), e-mail: claorisel@gmail.com.