УДК 519.171, 519.178

Б. Ф. Мельников, Е. Ф. Сайфуллина

ПРИМЕНЕНИЕ МУЛЬТИЭВРИСТИЧЕСКОГО ПОДХОДА ДЛЯ СЛУЧАЙНОЙ ГЕНЕРАЦИИ ГРАФА С ЗАДАННЫМ ВЕКТОРОМ СТЕПЕНЕЙ

Аннотация. *Актуальность и цели.* Графы с заданным вектором степеней часто рассматриваются в качестве моделей для многих сложных реальных задач. Это обусловливает актуальность исследования алгоритмов генерации графов с заданным вектором степеней. Целью исследования является рассмотрение существующих методов и разработка собственного алгоритма, позволяющего сгенерировать граф на основе заданного вектора степеней. Приводится определение графической последовательности; формулируются известные критерии проверки, является ли данная последовательность графической. *Результаты.* Приводится разработанный авторами алгоритм, представляющий собой реализацию одного из критериев проверки на основе мультиэвристического подхода (незавершенного метода ветвей и границ). Вводится понятие вектора степеней второго порядка и приводится модификация разработанного алгоритма на случай генерации графов с заданным вектором степеней второго порядка. Эта модификация также выполнена на основе незавершенного метода ветвей и границ. Таким образом, предлагается новый подход к случайной генерации графов как к задаче дискретной оптимизации. В ходе вычислительных экспериментов на основе некоторых функций распределения были сгенерированы последовательности заданного размера (предполагаемое число вершин графа). В случае если сгенерированная последовательность является графической, на ее основе может быть сгенерирован граф. Затем на основе вектора степеней второго порядка полученного графа был сгенерирован еще один граф. Приводятся результаты измерения среднего времени выполнения программы (в миллисекундах) в случае разных функций распределения и разных размерностей графа. *Выводы.* Приведенные разные варианты генерации случайных графов могут быть полезны во многих приложениях, прежде всего в сетевых моделях; среди последних наиболее важными являются математические модели Интернета и социальных сетей, а также модели функционирования искусственных нейронных сетей. Еще одним из возможных направлений продолжения работ, описанных в данной статье, является «настройка» конкретных алгоритмов решения задач дискретной оптимизации (в частности, задачи проверки изоморфизма на конкретные предметные области применения графов).

Ключевые слова: алгоритмы генерации случайных графов, вектор степеней и его обобщения, графическая последовательность, незавершенный метод ветвей и границ.

B. F. Mel'nikov, E. F. Sayfullina

APPLYING MULTIHEURISTIC APPROACH O RANDOMLY GENERATING GRAPHS WITH A GIVEN DEGREE SEQUENCE

Abstract. *Background.* Graphs with a given degree sequence are often regarded as models for many complex tasks. This makes it necessary to study algorithms for generating graphs with a given degree sequence. The purpose of this study is to re-

view the existing methods and to develop our own algorithm for generating a graph with a given degree sequence. The authors give the definition of a graphic sequence, as well as the criteria to check whether a given sequence is graphic or not. *Results*. The paper presents the algorithm developed by the authors which is based on one of the screening criteria of the multiheuristic approach (the branch and bound method). The paper also presents the notion of the second-order degree sequence and the modification of the developed algorithm for generating random graphs with a given second-order degree sequence. This modification is also based on the branch and bound method. Thus, a new approach to the generation of random graphs has been proposed. In the computational experiments based on some distribution functions the sequence of a given size (the predicted number of graph nodes) was generated. If the generated sequence is graphical, the graph can be generated on its basis. Then, another graph based on second-order degree sequence was generated. The average execution time (in milliseconds) within the different distribution functions and different dimensions of the graph was stated. *Conclusions.* These different options for generating random graphs can be useful in many applications, especially in network models, the most important of them being mathematical models of the Internet and social networks, as well as artificial neural networks. Another possible direction for continuation of the research described in this paper is a "tuning" of specific algorithms for solving discrete optimization problems (e.g. the problem of testing isomorphism in specific areas of the graph application).

Key words: algorithms for generating random graphs, a degree vector and and its generalizations, graphic sequence, the branch and bound method.

Введение

Графы с заданным вектором степеней[1] можно рассматривать в качестве конкретных вариантов моделей для многих сложных реальных задач [1]. По мнению авторов данной статьи, среди них самую важную роль играют сетевые модели, а среди конкретных сетей и сетевых моделей упомянем только несколько:

– Интернет и самые разные описывающие его функционирование математические модели [2, 3];[2]

– многочисленные социальные сети (и их модели [4]);

– одноранговые (пи́ринговые) сети [5];[3]

– нейронные сети – биологические и искусственные.

Для анализа алгоритмов работы с графами (в частности, с представлениями сетей в виде графов) и практической проверки программ, соответствующих этим алгоритмам, необходимы:

– либо заранее созданные базы данных, причем обычно очень больших размеров, соответствующие разным предметным областям и разным размерностям графа;

– либо алгоритмы генерации графов.

[1] Определение степени вершины графа стандартное [1, гл. 2]. На его основе определяется вектор степеней (вершин), который часто рассматривается в порядке убывания значений его элементов.

[2] Очень интересна серия связанных между собой статей, опубликованных в журнале «Труды Московского физико-технического института (государственного университета)», 2012, том 4, № 1 (13). По видимому, среди них для нашей статьи наиболее важна работа [3].

[3] IEEE по этой тематике практически ежегодно проводит международные конференции – “International Conference on Peer-to-Peer Computing”, см. [5].

По этой причине часто возникает задача *случайной генерации* графов *с заранее определенным вектором степеней* – определенным пользователем или какой-либо программой. Возможные варианты случайной генерации на основе вектора степеней можно найти в [6], а некоторые альтернативные подходы к случайной генерации приведены в [7, 8].

В данной статье мы не будем рассматривать вопрос *об адекватности* сгенерированных моделей некоторой предметной области[1]; отметим лишь, что одним из возможных подходов к данной проблеме можно считать подход, описанный в [9] для другой (фактически – более узкой) предметной области: для недетерминированных конечных автоматов. Мы рассмотрим только возможные алгоритмы случайной генерации графов, причем как авторские модификации ранее описанных алгоритмов, так и новые алгоритмы, точнее – алгоритмы, созданные на новых принципах. Во всех случаях мы будем применять гибридные алгоритмы, т.е. рассматривать случайную генерацию графов в качестве *задачи дискретной оптимизации* и применять к ее решению описанный нами мультиэвристический подход [10, 11].

При этом для наших вариантов случайной генерации графа мы применяем специальный его инвариант, являющийся обобщением вектора степеней. По этому поводу сто́ит отметить, что для вектора степеней ранее в литературе были описаны и принципиально иные варианты его обобщений, см., например, [12], а также работы, процитированные в той статье.

1. Предварительные замечания

Вектор степеней – одно из наиболее важных понятий, необходимых для рассмотрения наших задач.

Вектор степеней (графическая последовательность) – последовательность $(d_1, d_2, ..., d_n)$ целых неотрицательных чисел такая, что существует граф, последовательность степеней вершин которого с ней совпадает.

Наиболее известные способы проверки графичности заданной последовательности основаны на критериях Эрдеша – Галлаи (Erdös – Gallai) и Гавела – Хакими (Havel – Hakimi) [6, 13]. Для формулировки этих критериев дадим сначала определение правильной последовательности.

Правильная последовательность – последовательность натуральных чисел длины n, удовлетворяющая следующим условиям:

1) $n-1 \geq d_1 \geq d_2 \geq ... \geq d_n$;

2) $\sum_{i=1}^{n} d_i$ – четное число.

Критерий Эрдеша – Галлаи. Правильная последовательность d является графической тогда и только тогда, когда для каждого k, где $1 \leq k \leq n-1$, выполнено неравенство

$$\sum_{i=1}^{k} d_i \leq k(k-1) + \sum_{i=k+1}^{n} \min\{k, d_i\}.$$

[1] Отметим также следующее. По мнению авторов настоящей статьи, ни один из вопросов, рассматривавшихся в вышеупомянутой *серии* статей журнала «Труды Московского физико-технического института», нельзя считать рассмотрением адекватности предлагаемых моделей.

Критерий Гавела – Хакими. Пусть d – правильная последовательность. Зафиксируем индекс i, $1 \le i \le n$, и образуем последовательность c^i вычеркиванием из d i-го члена, а также последовательность d^i уменьшением на 1 первых d_i членов в c^i. Назовем d^i производной последовательностью. Если для данной последовательности d найдется такое i, что производная последовательность d^i является графической, то и d – графическая. Если d – графическая последовательность, то каждая последовательность d^i является графической [6].

Мультиэвристический подход к задачам дискретной оптимизации (МЭП) был подробно описан в нескольких работах одного из авторов – [10, 11] и др. Кроме того, описанную в [14] кластеризацию ситуаций, выполняемую в практических программах при работе МЭП, также можно рассматривать как предметную область появления случайных графов: число кластеров в примерах программ, реализованных на основе приведенных в [14] алгоритмов, на современной вычислительной технике превышает 1000. Отметим еще, что в недавней работе [15] фактически тот же самый подход применялся для реализации параллельных алгоритмов решения одной из задач дискретной оптимизации.

МЭП фактически представляет собой развитие метода ветвей и границ [16, 17], однако при этом к обычным вариантам его реализации добавляется реализация нескольких групп эвристик, дающих в комплексе убыстрение работы практических программ. Однако при этом очень важно отметить следующее. Ранее МЭП, как и следует из его названия, применялся нами к задачам дискретной оптимизации; и может показаться, что случайная генерация графов *оптимизационной* задачей не является. Однако мы все же рассматриваем ее именно таким образом – как задачу *восстановления* графа по заданному вектору, и при этом обычно целью (оптимальным решением) считается любое подходящее решение.

2. Существующие алгоритмы генерации графов с заданным вектором степеней и их модификации

Рассмотрим некоторые известные алгоритмы генерации графов с заданным вектором степеней [4, 6–8, 13, 18, 19]. Специально отметим, что мы приводим наши модификации ранее описанных алгоритмов, однако сами эти модификации не очень значительные, поэтому мы не всегда отмечаем конкретные изменения.

Во всех алгоритмах на вход подается вектор степеней $d = (d_1, ..., d_n)$; на основе, например, критерия Гавела – Хакими может быть проверена корректность входных данных.

Алгоритм, основанный на теореме Гавела – Хакими. Критерий Гавела – Хакими [13] дает следующий алгоритм для получения случайно сгенерированного графа с определенным вектором степеней d. В начале работы алгоритма задается множество вершин 1, ..., n и пустое множество ребер. На каждом шаге выбирается вершина i такая, что $d_i > 0$, при этом вероятность выбора вершин зависит от ее степени (т.е. от соответствующего значения в векторе d). В описанных в литературе алгоритмах эта зависимость обычно стро-

ится на основе равномерного распределения – мы же применяем и другие варианты распределений генерируемых случайных величин (см. об этом также в заключении). Соединяем эту вершину *i* ребрами с *i* вершинами, выбранными *аналогичным образом* (аналогично выбору вершины *i*). Далее из *d* получаем d', положив значение в *i*-й позиции равным 0 и уменьшая на 1 значения вектора, соответствующие выбранным *i* вершинам. Если полученная последовательность d' удовлетворяет сформулированным в предыдущем разделе критериям, то устанавливаем $d = d'$, в противном случае выбираем другое множество вершин для соединения ребрами с вершиной *i*.

Алгоритм, основанный на цепи Маркова и методе Монте-Карло. Широко применяемый подход – использование алгоритма, основанного на цепи Маркова и методе Монте-Карло (Markov chain Monte Carlo, MCMC algorithm [18]). Граф *G*, полученный на основе этой последовательности, задает начальное состояние цепи Маркова. Затем случайным образом выбираются два ребра $\{x,y\}$ и $\{u,v\}$, где x,y,u,v – четыре различные вершины. Если не существует ребер $\{x,u\}$ и $\{y,v\}$, то получаем граф G', добавив ребра $\{x,u\}$ и $\{y,v\}$ и удалив ребра $\{x,y\}$ и $\{u,v\}$. В противном случае остаемся в текущем состоянии. В качестве альтернативы можно также проверять, что не существует ребер $\{x,v\}$ и $\{y,u\}$. Если это условие выполняется и предыдущая замена ребер невозможна, то используем замену на эти ребра. Если же обе замены ребер возможны, то выбираем одну из них с вероятностью 0,5.

Алгоритмы, основанные на моделях с выбором пары вершин. Наиболее простые алгоритмы основаны на последовательном выборе пар вершин. Изначально задается пустое множество ребер. На каждом шаге с равной вероятностью выбирается пара вершин i и j, для которых еще возможно добавление ребра. Эти две вершины соединяются ребром. Процесс останавливается, когда больше не может быть добавлено ни одно ребро. Стегером и Вормалдом (Steger, Wormald [19]) был предложен вариант этого алгоритма с ограничениями на добавление петель и нескольких ребер из одной вершины в другую. Была показана эффективность этого алгоритма для случая однородных графов (степени всех вершин в однородном графе равны). В последующем была показана эффективность этого алгоритма для произвольных графов в случае, если пара вершин выбирается с вероятностью, равной

$$d_i d_j \left(1 - \frac{d_i d_j}{4m}\right),$$

где $\sum_{i=1}^{n} d_i = 2m$ и d_i – число ребер, которые можно добавить для вершины i.

Последовательные алгоритмы для построения графов и деревьев с заданным вектором степеней. Введем следующие обозначения:

$$(\oplus_{i_1,\ldots,i_k} d)_i = \begin{cases} d_i + 1 & \text{для } i \in \{i_1,\ldots,i_k\}, \\ d_i & \text{в противном случае;} \end{cases}$$

$$(\ominus_{i_1,\ldots,i} d)_i = \begin{cases} d_i - 1 & \text{для } i \in \{i_1,\ldots,i_k\}, \\ d_i & \text{в противном случае.} \end{cases}$$

Опишем сам алгоритм (отметим еще раз, что вход – $d = (d_1, ..., d_n)$):

1. E – пустое множество ребер.

2. Если все элементы последовательности d равны 0, то завершить алгоритм с множеством E *на выходе.*

3. Выбрать *последнюю* вершину i, для которой $d_i > 0$.

4. Составить список возможных вершин:

$J = \{ j \neq i : \{i, j\} \notin E$ и $\ominus_{i,j} d$ – графическая последовательность$\}$.

5. Выбрать вершину $j \in J$ с вероятностью, пропорциональной ее степени в векторе d.[1]

6. Добавить ребро $\{i, j\}$ к множеству E и положить d равным $\ominus_{i,j} d$.

7. Повторить шаги 4–6 до тех пор, пока степень i не станет равной 0.

8. Перейти к шагу 2.

Также в [19] была предложена модификация этого алгоритма на случай генерации *деревьев* с заданным вектором степеней. В этом случае на графическую последовательность налагаются дополнительные ограничения:

$$\sum_{i=1}^{n} d_i = 2n - 2 \quad \text{и} \quad d_i \geq 1.$$

1. E – пустое множество ребер.

2. Если элементы последовательности d равны 0, кроме $i \neq j$, где $d_i = d_j = 1$, то добавить ребро $\{i, j\}$ к множеству E и завершить алгоритм с множеством E *на выходе*.

3. Выбрать последнюю вершину i, для которой $d_i = 1$.

4. Выбрать вершину j, степень которой не меньше 2, с вероятностью, пропорциональной $d_j - 1$.[2]

5. Добавить ребро $\{i, j\}$ к множеству E и положить d равным $\ominus_{i,j} d$.

6. Перейти к шагу 2.

3. Генерация случайных графов как задача дискретной оптимизации

В данном разделе описывается принципиально иной подход к случайной генерации графа с заданным вектором степеней. А именно данная проблема рассматривается *как задача дискретной оптимизации* [17], в которой требуется *восстановить* граф по заданному вектору степеней. При этом, опять же по терминологии [17], любое найденное подходящее решение объявляется оптимальным (т.е. соответствующим образом выбирается целевая функция).

Из-за ограничений на объем статьи мы опускаем строгое формальное описание постановки задачи: на основе вышеизложенного это описание делается очевидным образом. Как было сказано выше, решение этой задачи было

[1] Как и ранее, в данном описании фактически рассматривается равномерное распределение, а авторы рассматривали и другие варианты. См. об этом в заключении.

[2] Примечание аналогично предыдущей сноске.

нами выполнено с использованием МЭП; приведем краткое описание решения (схему алгоритма), используя терминологию [11, 14].

Описание каждой подзадачи включает уже выбранные ребра графа – каждое из которых является также элементом множества *выбранных* разделяющих элементов; также описание подзадачи включает множество ребер, являющихся *табуированными* разделяющими элементами. Стартовая задача алгоритма – два пустых множества ребер в качестве выбранных и разделяющих элементов. На каждом шаге алгоритма (рис. 1) новый разделяющий элемент выбирается на основе одного из алгоритмов, рассмотренных в предыдущем разделе; на рис. 1 в качестве примера приведен выбор на основе наиболее простого из этих алгоритмов.

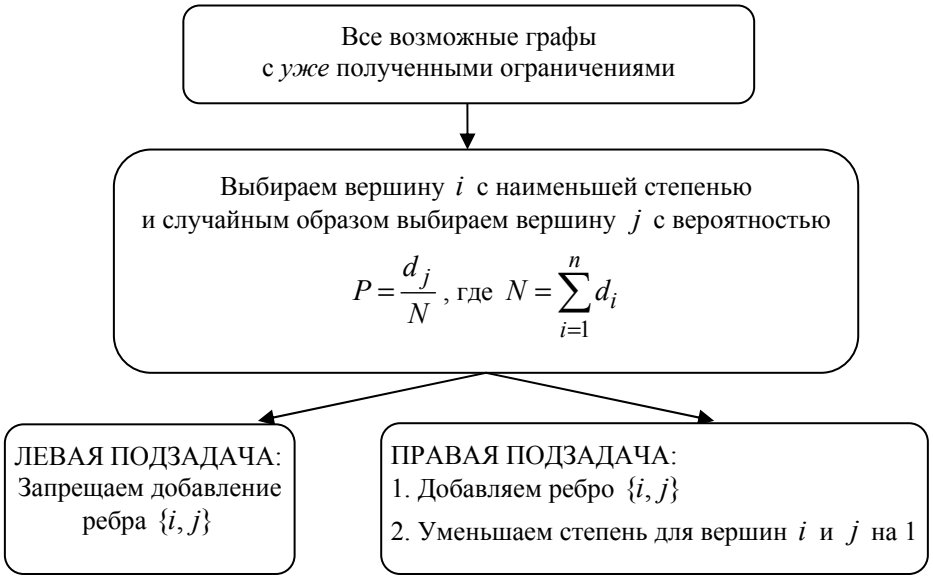

Рис. 1. Схема предлагаемого алгоритма

Итак, мы фактически описали реализацию *любого* алгоритма из предыдущего раздела в виде метода ветвей и границ; согласно [11] этот алгоритм мы можем назвать незавершенным, поскольку поиск всех подходящих решений нами не производится. Однако вряд ли описанная реализация алгоритмов предыдущего раздела представляла бы интерес, если бы не следующее обстоятельство, уже упомянутое во введении. Развитием данного подхода является применение того же самого алгоритма для значительно более сложной модели, рассматриваемой нами далее.

4. Вектор степеней второго порядка и подход к случайной генерации графа на его основе

Введем еще один инвариант (характеристику) графа – будем называть его вектором степеней второго порядка; заранее отметим, что подобный инвариант может быть построен, например, на основе более общих моделей, рассматривавшихся в [3]. Каждый элемент этого вектора представляет собой *список степеней вершин, смежных с данной вершиной*. Например, для графа,

представленного на рис. 2, вектор степеней второго порядка таков:

$$((5,4,3,3,2),(5,4,3,2,2),(5,5,3,2),(5,5,3),(5,4,3),(5,5),(5,4))$$

(при этом для упорядочивания вершин мы пользуемся естественным вариантом линейного порядка). Очевидно, что данная характеристика является инвариантом графа.

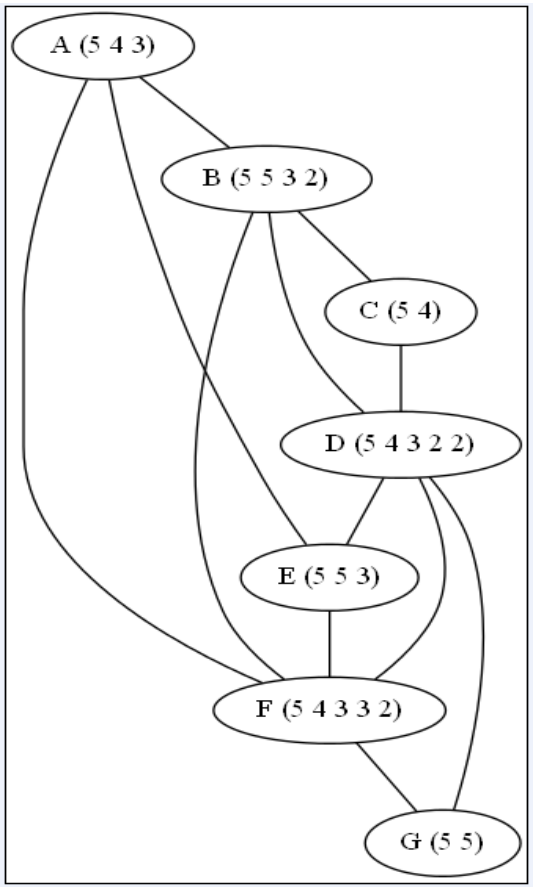

Рис. 2. Пример вычисления вектора степеней второго порядка для графа

Мы не будем рассматривать условия, описывающие, в каком случае заданный список списков является вектором степеней второго порядка некоторого графа; соответствующие условия – предмет отдельных публикаций. Отметим лишь, что несложные *необходимые* условия получаются на основе вышеприведенных критериев Эрдеша – Галлаи и Гавела – Хакими.

Описание алгоритма генерации графов на основе вектора степеней второго порядка близко к вышеприведенному описанию генерации графов на основе вектора степеней (первого порядка, рис. 1). Однако, в отличие от алгоритма, приведенного в предыдущем разделе, здесь запрет будет накладываться не на добавление ребра, а на добавление *множества ребер*. Для этого вводится список списков *Forbidden*, *i*-й элемент которого *Forbidden*[*i*] представляет множество вершин, в которые *запрещается добавлять ребра* из

вершины *i*. Приведем более подробное, чем в предыдущем разделе, описание алгоритма; при этом рисунок, аналогичный рис. 1, опускаем из-за ограничения на объем статьи.

Вход: список списков

$$v = ((v_1^1,...,v_1^{d_1}),...,(v_n^1,...,v_n^{d_n})),$$

где d_i – степень *i*-й вершины, v_i^j – степень *j*-й вершины, смежной с *i*-й вершиной, $i \in (1,...,n)$, $j \in (1,...,d_i)$, соответствующая последовательность $d = (d_1,...,d_n)$ графическая. Через $v[i]$ будем обозначать список, соответствующий *i*-й вершине.

1. *TaskList* – пустой список подзадач.

2. Инициализируем стартовую подзадачу предполагаемым вектором степеней второго порядка *v*, пустым списком списков запрещенных вершин *Forbidden*, пустым множеством ребер *E*, после чего добавляем ее в *TaskList*.

3. Если в первой лежащей в *TaskList* подзадаче для *v* все элементы списков равны 0, то завершаем алгоритм с множеством *E* этой подзадачи на выходе. Если же *TaskList* при этом пустой, то завершаем алгоритм с пустым множеством на выходе.

4. Для первой лежащей в *TaskList* подзадачи на основе ее списка списков *v* выберем последнюю вершину i, для которой в *v* соответствует список $v[i]$ наименьшего размера, состоящий из наименьших по значению ненулевых элементов.

5. Для всех $v_i^j \in v[i]$ таких, что $v_i^j \neq 0$, выберем вершину j с равной вероятностью из списка возможных вершин таких, что число элементов в $v[j]$ равно v_i^j, $j \notin Forbidden[i]$, при этом $v[j]$ содержит элемент, равный числу элементов в $v[i]$. Обозначим множество таких вершин *Chosen*.

6. Если выбрать вершины на предыдущем шаге не удалось, то удаляем данную подзадачу из списка *TaskList* и переходим к шагу 3.

7. Заменяем текущую подзадачу на две новые подзадачи. Левую подзадачу получаем, добавив вершины из множества *Chosen* в $Forbidden[i]$. Для формирования правой подзадачи для всех вершин из множества *Chosen* добавляем в *E* ребра из вершины *i* в эти вершины; при этом изменяем *v* следующим образом: для вершины *i* все элементы в $v[i]$ устанавливаем равными 0, для каждой вершины *j* из множества *Chosen* в $v[j]$ устанавливаем равным 0 элемент, равный числу элементов в $v[i]$. В списке *TaskList* левая подзадача заменяет текущую подзадачу, а правая подзадача добавляется в начало списка.

8. Переходим к шагу 3.

Выход: множество ребер *E*.

Итак, разработанные авторами алгоритмы генерации графов с заданными векторами степеней (первого и второго порядков) основаны на мультиэвристическом подходе к решению задач дискретной оптимизации. Причем в данном случае этот подход был применен не для получения какого-либо оптимального решения, а для *случайной* генерации.

5. Организация вычислений и результаты

Как уже было отмечено выше, в данной статье мы не рассматриваем вопрос об адекватности сгенерированных графов в некоторой предметной области. В связи с этим не может не возникнуть вопрос об организации вычислений для проверки эффективности вышеописанных эвристических алгоритмов [9, 20][1]. Мы их организовывали следующим образом, но, конечно же, в будущем необходимы более сложные вычислительные модели.

Вход: размерность и функция распределения значений вектора первого порядка.

1. На основе размерности и функции распределения генерируем вектор степеней первого порядка.

2. Проверяем сгенерированный вектор на критерии, описанные в разделе 2. Если этим критериям вектор не удовлетворяет, то возвращаемся на шаг 1.[2]

3. На основе вектора первого порядка мы *каким-либо* из вышеописанных алгоритмов (разделы 3, 4) генерируем *некоторый* граф, соответствующий вектору.

4. Для полученного графа генерируем соответствующий ему *вектор второго порядка*.

5. По заданному вектору 2-го порядка восстанавливаем сам граф – с помощью алгоритма, приведенного в разделе 5.[3]

Выход: множество ребер E.

В ходе вычислительных экспериментов на основе трех функций распределения были сгенерированы последовательности заданного размера (предполагаемое число вершин графа). В случае если сгенерированная последовательность является графической, на ее основе может быть сгенерирован граф. Затем на основе вектора степеней второго порядка полученного графа был сгенерирован еще один граф. Было измерено среднее время выполнения программы (в миллисекундах) в случае разных функций распределения и разных размерностей графа. Результаты эксперимента представлены в табл. 1.[4]

Заключение

Итак, в данной работе рассматривается обобщение одного из исследованных ранее инвариантов графа – вектора степеней – и описание примене-

[1] При этом необходимы два «противоположных друг другу» замечания. Во-первых, после описания специальных *характеристик* графов, *соответствующих некоторой предметной области* (аналогично тому, что сделано в [9] для конечных автоматов), мы должны каким-то образом *проверять* эти характеристики для сгенерированных нами графов. Во-вторых, наоборот: сами конкретные *алгоритмы генерации* графов могут быть изменены нами для *улучшения* требуемых характеристик («подгонка под ответ»). Сам *подход* изложен в [9] и, значительно более подробно, в [20].

[2] При этом во избежание зацикливания на данном шаге нами предусмотрен несложный механизм проверки, возможно ли вообще применение заданной нами функции распределения.

[3] Никакой другой информации мы при этом не используем. Отметим, что используемый вектор второго порядка при этом является *возможным* входом алгоритма генерации, поскольку он относится к некоторому *уже существующему* графу.

[4] По-видимому, конкретные характеристики вычислительного устройства и описание самой программы вряд ли интересны: они были одинаковы для всех 12 вариантов. Гораздо важнее, что для каждого из 12 вариантов было проведено по 100 вычислительных экспериментов, и во всех случаях при применении МЭП (точнее, описанной в разделе 5 его модификации) решение *было получено*.

ния для случайной генерации соответствующих графов мультиэвристического подхода к задачам дискретной оптимизации. Поскольку в данной статье мы не рассматриваем вопрос об адекватности сгенерированных графов в некоторой предметной области, основное направление в продолжение данной тематики связано с исследованием адекватности модели (репрезентативности); см. разделы 3 и 6 (в частности, сноски), и отметим при этом, что в них мы уже начали описывать возможные характеристики.

Однако имеется еще одно возможное направление продолжения работ, описанных в данной статье, это – «настройка» конкретных алгоритмов решения задач дискретной оптимизации (в частности, задачи проверки изоморфизма, [21]) на конкретные предметные области применения графов. Важно отметить, что *оба* этих направления дальнейших работ уже начали выполняться авторами для нескольких различных предметных областей. Среди них упомянем графы, возникающие в задачах проверки равенства бесконечных итераций конечных языков, см. [22].[1] Полученные результаты предполагается опубликовать в нескольких дальнейших работах.

Таблица 1

Результаты вычислительного эксперимента

Функция распределения	Размерность графа			
	7	14	21	28
Распределение Пуассона $p(k)=\frac{\lambda^k}{k!}e^{-\lambda},\ k=2,5$	94	110	406	610
Биноминальное распределение $P(k)=\left(\frac{n-1}{k}\right)p^k(1-p)^{n-1-k},\ p=0,25$	125	563	2344	10985
Распределение Ципфа $f(k,s,N)=\frac{1/k^s}{\sum_{n=1}^{N}(1/n^s)}$, $s=2,5,\ N-$ размерность графа	78	141	469	516

Список литературы

1. **Харари, Ф.** Теория графов / Ф. Харари. – М. : Мир, 1973. – 302 с.
2. **Райгородский, А. М.** Математические модели Интернета / А. М. Райгородский // Квант. – 2012. – № 4. – С. 12–16.
3. **Остроумова, Л. А.** Математические ожидания *k*-х входящих степеней вершин в случайных графах в модели Боллобаша-Риордана / Л. А. Остроумова // Труды Московского физико-технического института. – 2012. – Т. 4, № 1 (13). – С. 29–40.
4. **Бреер, В. В.** Стохастические модели социальных сетей / В. В. Бреер // Управление большими системами. – 2009. – № 27. – С. 169–204.

[1] Вряд ли можно надеяться, что для этих проблем в ближайшем будущем появятся соответствующие базы данных графов, описывающих рассматриваемые языки. Поэтому все практические алгоритмы, созданные на основе [22], уже тестировались авторами с помощью алгоритмов случайной генерации графов, близких к описываемым в данной статье.

5. URL: http://www.p2p13.org/ (дата обращения: 16.06.2013).
6. **Erdös, P.** A Simple Havel-Hakimi Type Algorithm to Realize Graphical Degree Sequences of Directed Graphs / P. Erdös, I. Miklós, Z. Toroczkai // Electr. J. Comb. – 2010. – V. 17. – № 1.
7. **Bollobás B.** Random Graphs / B. Bollobás. – Cambridge : Cambridge Univ. Press, 2001.
8. **Bollobás, B.** Mathematical results on scale-free random graphs / B. Bollobás, O. Riordan // Handbook of graphs and networks. – Weinheim : Wiley-VCH, 2003. – P. 1–34.
9. **Мельников, Б. Ф.** Репрезентативность случайно сгенерированных недетерминированных конечных автоматов с точки зрения соответствующих базисных автоматов / Б. Ф. Мельников, С. Пивнева, О. Рогова // Стохастическая оптимизация в информатике. – 2010. – № 6. – С. 74–82.
10. **Melnikov, B.** Discrete optimization problems – some new heuristic approaches / B. Melnikov // Proc. of the Eighth International Conference on High-Performance Computing in Asia-Pacific Region, IEEE Computer Society. – Washington, 2005. – P. 73–80.
11. **Мельников, Б. Ф.** Мультиэвристический подход к задачам дискретной оптимизации / Б. Ф. Мельников // Кибернетика и системный анализ (НАН Украины). – 2006. – № 3. – С. 32–42.
12. **Iványi. A.** Reconstruction of complete interval tournaments / A. Iványi // Acta Universitatis Sapientiae, Informatica. – 2009. – Vol. 1, № 1. – P. 71−88.
13. **Frank, H.** Circuit Theory / H. Frank, S. Hakimi // IEEE Transactions on. – 1965, Vol. 12, № 3. – P. 44–51.
14. **Мельников**, **Б. Ф.** Кластеризация ситуаций в алгоритмах реального времени в некоторых задачах дискретной оптимизации / Б. Ф. Мельников, Е. А. Мельникова // Известия высших учебных заведений. Поволжский регион. Естественные науки. – 2007. – № 6. – С. 3–11.
15. **Melnikov, B.** The state minimization problem for nondeterministic finite automata: the parallel implementation of the truncated branch and bound method / B. Melnikov, A. Tsyganov // 5th Int. Symp. on Parallel Architectures, Algorithms and Programming, IEEE Computer Society Ed. – Taipei, 2012. – P. 194–201.
16. **Гудман, С.** Введение в разработку и анализ алгоритмов / С. Гудман, С. Хидетниеми. – М. : Мир, 1981. – 368 с.
17. **Громкович, Ю.** Теоретическая информатика. Введение в теорию автоматов, теорию вычислимости, теорию сложности, теорию алгоритмов, рандомизацию, теорию связи и криптографию / Ю. Громкович. – СПб. : БХВ-Петербург, 2010. – 336 с.
18. **Berg, B.** Markov Chain Monte Carlo Simulations and Their Statistical Analysis / B. Berg. – Singapore, World Scientific Publ., 2004. – 361 p.
19. **Steger, A.** Generating random regular graphs quickly / A. Steger, N. Wormald // Combinatorics, Probab. and Comput. – 1999. – № 8. – P. 377–396.
20. **Рогова, О. А.** Подход к оценке репрезентативности случайно сгенерированных дискретных структур на примере недетерминированных конечных автоматов : дис. ... канд. физ.-мат. наук / Рогова О. А. – Тольятти : ТГУ, 2012. – 114 с.
21. **Мельникова, Е. А.** Применение различных инвариантов графов к проверке изоморфизма некоторых видов графов / Е. А. Мельникова, Е. Ф. Сайфуллина // Проблемы информатики в образовании, управлении, экономике и технике : тр. XII Междунар. научно-техн. конф. – Пенза : Приволжский Дом знаний, 2012. – С. 40–42.
22. **Мельников, Б. Ф.** Алгоритм проверки равенства бесконечных итераций конечных языков / Б. Ф. Мельников // Вестник Московского университета. Сер. Вычисл. матем. и киб-ка. – 1996. – № 4. – С. 49–54.

References

1. Kharari F. *Teoriya grafov* [Graph theory]. Moscow: Mir, 1973, 302 p.
2. Raygorodskiy A. *Kvant* [Quantum]. 2012, no. 4, pp. 12–16.
3. Ostroumova L. *Trudy Moskovskogo fiziko-tekhnicheskogo instituta* [Proceedings of Moscow Institute of Physics and Technology]. 2012, vol. 4, no. 1 (13), pp. 29–40.
4. Breer V. *Upravlenie bol'shimi sistemami* [Grand system control]. 2009, no. 27, pp. 169–204.
5. Available at: http://www.p2p13.org/ (accessed 16 June 2013).
6. Erdös P., Miklós I., Toroczkai Z. *Electr. J. Comb.* 2010, vol. 17, no. 1.
7. Bollobás B. *Random Graphs*. Cambridge: Cambridge Univ. Press, 2001.
8. Bollobás B., Riordan O. *Handbook of graphs and networks*. Weinheim: Wiley-VCH, 2003, pp. 1–34.
9. Mel'nikov B., Pivneva S., Rogova O. *Stokhasticheskaya optimizatsiya v informatike* [Stochastic optimization in computer science]. 2010, no. 6, pp. 74–82.
10. Melnikov B. *Proc. of the Eighth International Conference on High-Performance Computing in Asia-Pacific Region, IEEE Computer Society*. Washington, 2005, pp. 73–80.
11. Mel'nikov B. *Kibernetika i sistemnyy analiz (NAN Ukrainy)* [Cybernatics and system analysis (National Academy of Sciences of Ukraine)]. 2006, no. 3, pp. 32–42.
12. Iványi. A. *Acta Universitatis Sapientiae, Informatica*. 2009, vol. 1, no. 1, pp. 71−88.
13. Frank H., Hakimi S. *IEEE Transactions on*. 1965, vol. 12, no. 3, pp. 44–51.
14. Mel'nikov B., Mel'nikova E. *Izvestiya vysshikh uchebnykh zavedeniy. Povolzhskiy region. Estestvennye nauki* [University proceedings. Volga region. Natural sciences]. 2007, no. 6, pp. 3–11.
15. Melnikov B., Tsyganov A. *5th Int. Symp. on Parallel Architectures, Algorithms and Programming, IEEE Computer Society Ed*. Taipei, 2012, pp. 194–201.
16. Gudman S., Khidetniemi S. *Vvedenie v razrabotku i analiz algoritmov* [Introduction into algorithm design and analysis]. Moscow: Mir, 1981, 368 p.
17. Gromkovich Yu. *Teoreticheskaya informatika. Vvedenie v teoriyu avtomatov, teoriyu vychislimosti, teoriyu slozhnosti, teoriyu algoritmov, randomizatsiyu, teoriyu svyazi i kriptografiyu* [Theoretical computer sciences. Introduction into automata theory, theory of calculability, complexity theory, algorithm theory, randomization, communication and cryptography theory]. Saint Petersburg: BKhV-Peterburg, 2010.
18. Berg B. *Markov Chain Monte Carlo Simulations and Their Statistical Analysis*. Singapore, World Scientific Publ., 2004, 361 p.
19. Steger A., Wormald N. *Combinatorics, Probab. and Comput.* 1999, no. 8, pp. 377–396.
20. Rogova O. *Podkhod k otsenke reprezentativnosti sluchayno sgenerirovannykh diskretnykh struktur na primere nedeterminirovannykh konechnykh avtomatov: dis. kand. fiz.-mat. nauk* [Approach to estimation of randomly generated discrete structure representativeness by the example of nondeterministic finite automata: dissertation to apply for the degree of the candidate of physical and mathematical sciences]. Tolyatti: TGU, 2012.
21. Mel'nikova E., Sayfullina E. *Problemy informatiki v obrazovanii, upravlenii, ekonomike i tekhnike: tr. XII Mezhdunar. nauchno-tekhn. konf.* [Problems of computer science in education, administration, economy and technology: proceedings of XII International scientific technological conference]. Penza: Privolzhskiy Dom znaniy, 2012, pp. 40–42.
22. Mel'nikov B. *Vestnik Mosk. un-ta. Ceriya. Vychisl. matem. i kib-ka* [Bulletin of Moscow University. Series. Calculus mathematics and cybernetics]. 1996, no. 4, pp. 49–54.

Мельников Борис Феликсович
доктор физико-математических наук, профессор, кафедра прикладной математики и информатики, Тольяттинский государственный университет (Россия, г. Тольятти, ул. Белорусская, 14)

E-mail: B.Melnikov@tltsu.ru

Mel'nikov Boris Feliksovich
Doctor of physical and mathematical sciences, professor, sub-department of applied mathematics and informatics, Togliatti State University (14 Belorusskaya street, Togliatti, Russia)

Сайфуллина Елена Фаридовна
аспирант, Тольяттинский государственный университет (Россия, г. Тольятти, ул. Белорусская, 14)

E-mail: elena-fairy@yandex.ru

Sayfullina Elena Faridovna
Postgraduate student, Togliatti State University (14 Belorusskaya street, Togliatti, Russia)

УДК 519.171, 519.178

Мельников, Б. Ф.

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