



ИНФОРМАТИКА

УДК 519.17

Построение всех неизоморфных минимальных вершинных расширений графа методом канонических представителей

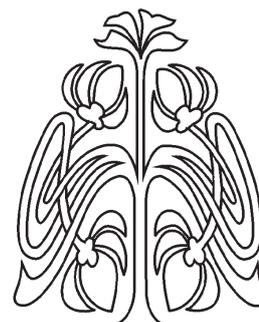
М. Б. Абросимов, И. А. К. Камил, А. А. Лобов

Абросимов Михаил Борисович, доктор физико-математических наук, заведующий кафедрой теоретических основ компьютерной безопасности и криптографии, Саратовский национальный исследовательский государственный университет имени Н. Г. Чернышевского, Россия, 410012, г. Саратов, ул. Астраханская, д. 83, mic@rambler.ru

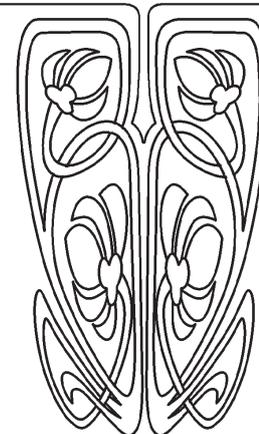
Камил Ихаб Абджулджаббар Камил, Министерство науки и технологий Ирака, Багдад, Ирак; аспирант кафедры теоретических основ компьютерной безопасности и криптографии, Саратовский национальный исследовательский государственный университет имени Н. Г. Чернышевского, Россия, 410012, г. Саратов, ул. Астраханская, д. 83, kamil.iehab@mail.ru

Лобов Александр Андреевич, аспирант кафедры теоретических основ компьютерной безопасности и криптографии, Саратовский национальный исследовательский государственный университет имени Н. Г. Чернышевского, Россия, 410012, г. Саратов, ул. Астраханская, д. 83, aisaneka@mail.ru

В 1976 г. John P. Hayes предложил основанную на графах модель для исследования отказоустойчивости дискретных систем. Технической системе сопоставляется граф. Элементам системы соответствуют вершины графа, а связям между элементами — рёбра или дуги графа. Под отказом элемента системы понимается удаление из графа системы соответствующей вершины вместе со всеми её рёбрами. Формализацией отказоустойчивой реализации системы является расширение графа. Граф G^* называется вершинным k -расширением графа G , если после удаления любых k вершин из графа G^* граф G вкладывается в получившийся граф. Вершинное k -расширение графа G называется минимальным, если оно имеет наименьшее число вершин и рёбер среди всех вершинных k -расширений графа G . В работе предлагается алгоритм построения всех неизоморфных минимальных вершинных k -расширений заданного графа без проверки на изоморфизм методом канонических представителей.



НАУЧНЫЙ
ОТДЕЛ





Ключевые слова: отказоустойчивость, расширение графа, изоморфизм, канонический код, метод канонических представителей.

Поступила в редакцию: 20.05.2019 / Принята: 30.06.2019 / Опубликовано: 02.12.2019

Статья опубликована на условиях лицензии Creative Commons Attribution License (CC-BY 4.0)

DOI: <https://doi.org/10.18500/1816-9791-2019-19-4-479-486>

ВВЕДЕНИЕ

В 1976 г. John P. Hayes [1] предложил основанную на графах модель для исследования отказоустойчивости дискретных систем. Технической системе сопоставляется граф. Элементам системы соответствуют вершины графа, а связям между элементами — рёбра или дуги графа. Под отказом элемента системы понимается удаление из графа системы соответствующей вершины вместе со всеми её рёбрами. Позднее совместно с Frank Nagay модель была распространена на отказы связей [2]. Формализацией отказоустойчивой реализации системы является расширение графа [3]. В работе рассматриваются неориентированные графы, а основные понятия используются согласно работе [4].

Граф $G^* = (V^*, \alpha^*)$ называется вершинным k -расширением (k — натуральное) графа $G = (V, \alpha)$, если граф G вкладывается в каждый подграф графа G^* , получающийся удалением любых его k вершин.

Граф $G^* = (V^*, \alpha^*)$ называется минимальным вершинным k -расширением (МВ- k Р) n -вершинного графа $G = (V, \alpha)$, если выполняются следующие условия:

- 1) граф G^* является вершинным k -расширением графа G ;
- 2) граф G^* содержит $n + k$ вершин;
- 3) α^* имеет минимальную мощность среди всех графов, удовлетворяющих условиям 1 и 2.

В работе [1] были предложены схемы построения МВ-1Р для цепей и циклов. В работе [5] было доказано, что задача построения вершинных k -расширений является вычислительно сложной. В этой статье будет рассмотрен переборный алгоритм построения всех МВ- k Р для заданного графа, а также алгоритмы построения всех неизоморфных МВ- k Р без проверки на изоморфизм. Мы будем рассматривать в основном случай $k = 1$, хотя все описанные методы работают и для общего случая. Напомним, что два графа $G_1 = (V_1, \alpha_1)$ и $G_2 = (V_2, \alpha_2)$ называются *изоморфными*, если можно установить взаимно однозначное соответствие $\phi: V_1 \rightarrow V_2$, сохраняющее отношение смежности: $(u, v) \in \alpha_1 \leftrightarrow (\phi(u), \phi(v)) \in \alpha_2, \forall u, v \in V_1$.

1. ПОСТРОЕНИЕ МИНИМАЛЬНЫХ ВЕРШИННЫХ РАСШИРЕНИЙ ГРАФА

Для построения всех неизоморфных МВ- k Р заданного графа в работе [3] был предложен общий переборный алгоритм:

Алгоритм 1. Построение всех МВ- k Р графа G .

1. $m := 0$.
2. $m := m + 1$.
3. Строим все графы, получающиеся из графа G добавлением k вершин и m дополнительных рёбер.
4. Выбираем среди построенных графов вершинные k -расширения графа G .
5. Если на шаге 4 не было найдено графов, то переходим на шаг 2.



6. Среди графов, выбранных на шаге 4, оставляем по одному представителю от классов изоморфных графов. Оставшиеся графы являются МВ- k Р графа G .

На практике шаги 3 и 4 могут быть совмещены. Рассмотрим идею работы алгоритма на примере построения МВ-1Р для 6-вершинного цикла C_6 .

На шаге 3 алгоритма необходимо добавить к циклу C_6 k вершин (в нашем случае $k = 1$) и m рёбер, где $m = 1, 2, \dots$. Такой граф можно записать как $C_6 \cup O_k$, где \cup обозначает операцию объединения графов, а O_k — k -вершинный вполне несвязный граф (в нашем случае — одна изолированная вершина).

Добавить $m = 1$ ребро к графу $C_6 \cup O_1$ означает заменить один из нулей, расположенных в матрице смежности выше главной диагонали, на единицу. При непосредственном переборе всех вариантов это можно сделать C_{15}^m , т. е. 15 способами. При этом некоторые получившиеся графы окажутся изоморфными. Легко заметить, что на самом деле получится только 3 неизоморфных графа. При $m = 2$ число способов будет уже 105, а неизоморфных графов будет только 15, при $m = 3$ — соответственно 455 и 48.

Далее каждый получившийся граф проверяется на шаге 4 — является ли он вершинным 1-расширением заданного графа. В общем случае для этого требуется проверить, что исходный граф допускает вложение в каждый граф, получающийся из очередного кандидата удалением одной вершины: может потребоваться $n + 1$ проверка. Некоторые оптимизации перебора можно сделать с использованием следующих лемм [3, 6].

Лемма 1. *Если минимальная степень вершины графа G есть $d > 0$, то его МВ- k Р G^* не содержит вершин степени ниже $d + k$.*

Лемма 2. *Пусть наибольшая из степеней вершин графа G есть s и в точности t вершин имеют такую степень, тогда МВ- k Р графа G содержит, по крайней мере, $k + t$ вершин степени не ниже s .*

Лемма 3. *Если граф G^* является вершинным k -расширением графа G и в графе G^* на t рёбер больше, чем в графе G , то для каждого набора (v_1, \dots, v_k) из k вершин графа G^* выполняется следующее неравенство: $d(v_1) + \dots + d(v_k) - r \leq t$, где r — количество рёбер в подграфе графа G^* , состоящем из вершин $\{v_1, \dots, v_k\}$.*

Например, с помощью леммы 1 для цикла C_6 можно установить, что в МВ-1Р не может быть вершин степени ниже 3. Как следствие, число дополнительных рёбер будет не меньше 5. В работе [1] доказывается, что МВ-1Р цикла имеет вектор степеней $(3, \dots, 3)$ или $(4, 3, \dots, 3)$ в зависимости от чётности числа вершин цикла. Число вариантов перебора графов при $m = 5$ составит 3003, из которых неизоморфных графов будет только 125, неизоморфных графов, удовлетворяющих лемме 1 (т. е. графов с вектором степеней $(4, 3^6)$), всего 4, а МВ-1Р цикла C_6 всего 2. Леммы 2 и 3 также нужно использовать для сокращения перебора. Алгоритм 1 можно использовать для построения МВ-1Р графов с небольшим числом вершин [7], однако его эффективность очень низкая. Наметим основной путь для оптимизации алгоритма 1: на шаге 3 нужно оставлять как можно меньше кандидатов для последующей проверки на расширение на шаге 4. Далее в статье будет рассмотрено решение этой задачи.



2. ГЕНЕРАЦИЯ БЕЗ ПРОВЕРКИ НА ИЗОМОРФИЗМ

У алгоритма 1 можно выделить несколько недостатков, связанных с избыточным перебором. Один из них состоит в следующем: если на шаге 3 могут появляться изоморфные графы, то необходимо хранить все построенные расширения, чтобы на шаге 6 исключить изоморфные копии. Если на шаге 3 строить только неизоморфные графы, то необходимость хранения всех построенных расширений исчезнет, а также число графов, которые нужно будет проверять на расширение, уменьшится. Методы генерации без проверки на изоморфизм достаточно интенсивно исследуются для построения графов различного вида. Хороший обзор можно найти в работе [8]. Далее в статье будет описана реализация метода канонических представителей для рассматриваемой задачи построения $MB-kP$.

Идея метода канонических представителей состоит в том, что выбирается некоторое правило, по которому из всех изоморфных графов один граф объявляется каноническим. В общем виде метод можно описать следующим образом [8]:

- 1) определяется способ кодирования графов;
- 2) среди всех кодов изоморфных графов выбирается канонический код (представитель);
- 3) порождаются все возможные коды графов;
- 4) порождённый граф принимается, если его код канонический, в противном случае исключается.

С использованием метода канонических представителей алгоритм 1 можно преобразовать в алгоритм 2.

Алгоритм 2. Построение всех $MB-kP$ графа G без проверки на изоморфизм.

1. $m := 0$.
2. $m := m + 1$.
3. Строим все неизоморфные графы, получающиеся из графа G добавлением k вершин и m рёбер.
4. Выбираем среди построенных графов вершинные k -расширения графа G .
5. Если на шаге 4 не было найдено графов, то переходим на шаг 2.
6. Полученные на шаге 4 графы являются $MB-kP$ графа G .

В алгоритме 2 при использовании метода канонических представителей не нужно проверять графы на изоморфизм. Для правильности предложенного алгоритма необходимо, чтобы все канонические представители каждого класса изоморфных графов были построены на шаге 3. Для использования метода канонических представителей самым важным является выбор собственно канонического кода. В данной работе предлагается использовать код особого вида, основанный на матрице смежности графа. Напомним, что для простых неориентированных графов матрица смежности симметрична относительно главной диагонали, а на главной диагонали расположены нули. Поэтому для кодирования матрицы смежности графа достаточно элементов матрицы смежности, расположенных выше главной диагонали.

Для начала определим способ кодирования графа. Рассмотрим два n -вершинных графа $G = (Z_n, \alpha)$ и $H = (Z_n, \beta)$, где $Z_n = \{0, \dots, n-1\}$ и $\alpha, \beta \subseteq Z_n \times Z_n$. Для каждого графа можно составить матрицу смежности, столбцы и строки которой помечены метками вершин $0, \dots, n-1$. На пересечении строки i и столбца j стоит 1 тогда и только тогда, когда в графе существует ребро $\{i, j\}$, иначе стоит 0.

Через G обозначим граф, для которого требуется найти $MB-kP$. Через H обозначим граф, для которого будем строить код. Добавим к графу G k изо-



лированных вершин. Определим код $C_G(H)$ графа H следующим образом: будем дважды просматривать элементы матрицы смежности графа G , находящиеся выше главной диагонали, по столбцам слева направо, и выписывать соответствующие элементы матрицы смежности графа H по следующим правилам:

- 1) при первом проходе выписываем элемент матрицы смежности H , если в матрице смежности G стоит 1;
- 2) при втором проходе выписываем элемент матрицы смежности H , если в матрице смежности G стоит 0.

В столбце элементы матрицы перечисляются сверху вниз. По построению очевидно, что код n -вершинного графа с m рёбрами будет иметь длину C_n^2 и содержать ровно m единиц.

На рисунке приведён пример построения кода.

	M_G	Порядок выписывания [Writing procedure]	M_H																																																																											
	<table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td>0</td><td><u>1</u></td><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>0</td><td><u>1</u></td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td><td><u>1</u></td><td><u>1</u></td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr> </table>	0	<u>1</u>	0	0	0	1	0	<u>1</u>	0	0	0	1	0	<u>1</u>	<u>1</u>	0	0	1	0	0	0	0	1	0	0	<table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td>0</td><td><u>1</u></td><td>5</td><td>6</td><td>8</td></tr> <tr><td>1</td><td>0</td><td><u>2</u></td><td>7</td><td>9</td></tr> <tr><td>0</td><td>1</td><td>0</td><td><u>3</u></td><td>4</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>10</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr> </table>	0	<u>1</u>	5	6	8	1	0	<u>2</u>	7	9	0	1	0	<u>3</u>	4	0	1	0	0	10	0	0	1	1	0	<table border="1" style="border-collapse: collapse; text-align: center;"> <tr><td>0</td><td><u>1</u></td><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>0</td><td><u>1</u></td><td><u>1</u></td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td><td><u>0</u></td><td><u>1</u></td></tr> <tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr> </table>	0	<u>1</u>	0	0	0	1	0	<u>1</u>	<u>1</u>	0	0	1	0	<u>0</u>	<u>1</u>	0	1	0	0	1	0	0	1	1	0
0	<u>1</u>	0	0	0																																																																										
1	0	<u>1</u>	0	0																																																																										
0	1	0	<u>1</u>	<u>1</u>																																																																										
0	0	1	0	0																																																																										
0	0	1	0	0																																																																										
0	<u>1</u>	5	6	8																																																																										
1	0	<u>2</u>	7	9																																																																										
0	1	0	<u>3</u>	4																																																																										
0	1	0	0	10																																																																										
0	0	1	1	0																																																																										
0	<u>1</u>	0	0	0																																																																										
1	0	<u>1</u>	<u>1</u>	0																																																																										
0	1	0	<u>0</u>	<u>1</u>																																																																										
0	1	0	0	1																																																																										
0	0	1	1	0																																																																										
	$C_G(G) = $ 1111000000		$C_G(H) = $ 1101001001																																																																											

Пример построения кода [Sample of code construction]

Будем называть граф H каноническим относительно G (либо просто каноническим) и его код каноническим, если среди всех графов, изоморфных H , код графа H является лексикографически наибольшим:

$$\{\forall R \cong H, R \neq H : C_G(R) < C_G(H)\}.$$

Справедливы следующие утверждения:

Утверждение 1. Если граф G является частью графа H , то $C_G(G) \leq C_G(H)$, иначе $C_G(G) > C_G(H)$.

Утверждение 2. Граф G вкладывается в граф H тогда и только тогда, когда существует $W \cong H$ такой, что $C_G(W) \geq C_G(G)$. Это означает, что если граф G вкладывается в граф H , то существует изоморфный ему канонический граф W : $C_G(W) \geq C_G(G)$.

Это значит, что канонического представителя класса изоморфизма каждого графа, в который вкладывается граф G , можно получить добавлением рёбер в граф G . Таким образом, алгоритм 2 является корректным.

Алгоритм 3. Построение всех неизоморфных графов, отличающихся от заданного графа G на m дополнительных рёбер, методом канонических представителей.

1. Начальным выбирается код $c = 1^p 0^{C_n^2 - p}$.



2. Перебираем все варианты размещения m единиц в хвосте из 0 кода s . Если очередной получившийся код s' является каноническим, то добавляем s' к списку-результату.

Обычно при реализации метода канонических представителей нет необходимости перебирать всех возможных представителей. Если на некотором шаге известно, что последующие итерации не могут привести к каноническому представителю, то можно произвести усечение перебора. Такая оптимизация называется методом Рида – Фараджева. В нашем примере построение графов можно рассматривать как перебор кодов, которые начинаются с кода графа G и отличаются от него добавлением m единиц в тех позициях, где стоят нули. Дополнительные отсечения можно сделать и на основе лемм 1–3.

На языке C++ с использованием MPI были реализованы алгоритмы 1 и 2. Был произведён вычислительный эксперимент по построению минимальных вершинных 1- и 2-расширений всех графов с числом вершин до 9 и минимальных вершинных 1-расширений циклов с числом вершин до 20. Для построения всех графов с заданным числом вершин использовался генератор geng из пакета nauty [9]. Вычисления выполнялись на кластере Поволжского регионального центра новых информационных технологий (ПРЦНИТ) [10]. В вычислениях было задействовано 16 ядер. Полученные в рамках вычислительного эксперимента результаты были добавлены в энциклопедию «Мир графов» [11]. Можно отметить существенный прирост в скорости построения МВ-кР для алгоритма 2 по сравнению с алгоритмом 1.

Библиографический список

1. Hayes J. P. A graph model for fault-tolerant computing system // IEEE Transactions on Computers, 1976. Vol. C-25, iss. 9. P. 875–884. DOI: <https://doi.org/10.1109/TC.1976.1674712>
2. Harary F., Hayes J. P. Edge fault tolerance in graphs // Networks. 1993. Vol. 23. P. 135–142. DOI: <https://doi.org/10.1002/net.3230230207>
3. Абросимов М. Б. Графовые модели отказоустойчивости. Саратов : Изд-во Сарат. ун-та, 2012. 192 с.
4. Богомолов А. М., Салий В. Н. Алгебраические основы теории дискретных систем. М. : Наука, 1997. 368 с.
5. Абросимов М. Б. О сложности некоторых задач, связанных с расширениями графов // Матем. заметки. 2010. Т. 88, вып. 5. С. 643–650. DOI: <https://doi.org/10.4213/mzm8403>
6. Абросимов М. Б. Минимальные расширения графов // Новые информационные технологии в исследовании дискретных структур : докл. Третьей Всерос. конф. с междунар. участием. Томск, 2000. С. 59–64.
7. Абросимов М. Б. Минимальные расширения 4-, 5-, 6- и 7-вершинных графов. Саратов. гос. ун-т. Саратов, 2000. 26 с.; Деп. в ВИНТИ 06.09.2000, № 2352-B00.
8. Brinkmann G. Isomorphism rejection in structure generation programs // DIMACS Series in Discrete Mathematics and Theoretical Computer Science. 2000. Vol. 51. P. 25–38. DOI: <https://doi.org/10.1090/dimacs/051/03>
9. McKay B. D., Piperno A. Practical Graph Isomorphism, II // Journal of Symbolic Computation. 2014. Vol. 60. P. 94–112. DOI: <https://doi.org/10.1016/j.jsc.2013.09.003>
10. Поволжский региональный центр новых информационных технологий. URL: <http://prcnit.sgu.ru> (дата обращения: 01.05.2019).
11. Мир графов. URL: <http://graphworld.ru> (дата обращения: 01.05.2019).



Образец для цитирования:

Абросимов М. Б., Камил И. А. К., Лобов А. А. Построение всех неизоморфных минимальных вершинных расширений графа методом канонических представителей // Изв. Саратов. ун-та. Нов. сер. Сер. Математика. Механика. Информатика. 2019. Т. 19, вып. 4. С. 479–486. DOI: <https://doi.org/10.18500/1816-9791-2019-19-4-479-486>

Construction of All Nonisomorphic Minimal Vertex Extensions of the Graph by the Method of Canonical Representatives

M. B. Abrosimov, I. A. K. Kamil, A. A. Lobov

Mikhail B. Abrosimov, <https://doi.org/0000-0002-4473-8790>, Saratov State University, 83 Astrakhanskaya St., Saratov 410012, Russia, mic@rambler.ru

Iehab A. K. Kamil, <https://doi.org/0000-0003-1100-5635>, Ministry of Science and Technology of Iraq, Baghdad, Iraq; Saratov State University, 83 Astrakhanskaya St., Saratov 410012, Russia, kamil.iehab@mail.ru

Alexander A. Lobov, <https://doi.org/0000-0003-3422-3811>, Saratov State University, 83 Astrakhanskaya St., Saratov 410012, Russia, aisaneka@mail.ru

In 1976 John P. Hayes proposed a graph model for investigating the fault tolerance of discrete systems. The technical system is mapped to a graph. The elements of the system correspond to the vertices of the graph, and links between the elements correspond to edges or arcs of the graph. Failure of a system element refers to the removal of the corresponding vertex from the system graph along with all its edges. Later together with Frank Harary the model was extended to links failures. The formalization of a fault-tolerant system implementation is the extension of the graph. The graph G^* is called the vertex k -extension of the graph G if after removing any k vertices from the graph G^* result graph contains the graph G . A vertex k -extension of the graph G is called minimal if it has the least number of vertices and edges among all vertex k -extensions of the graph G . An algorithm for constructing all nonisomorphic minimal vertex k -extensions of the given graph using the method of canonical representatives is proposed.

Keywords: fault tolerance, isomorphism rejection, canonical form, graph extension.

Received: 20.05.2019 / Accepted: 30.06.2019 / Published: 02.12.2019

This is an open access article distributed under the terms of Creative Commons Attribution License (CC-BY 4.0)

References

1. Hayes J. P. A graph model for fault-tolerant computing system. *IEEE Transactions on Computers*, 1976, vol. C-25, iss. 9, pp. 875–884. DOI: <https://doi.org/10.1109/TC.1976.1674712>
2. Harary F., Hayes J. P. Edge fault tolerance in graphs. *Networks*, 1993, vol. 23, pp. 135–142. DOI: <https://doi.org/10.1002/net.3230230207>
3. Abrosimov M. B. *Grafovyye modeli otkazoustoichivosti* [Fault tolerance graph models]. Saratov, Izd-vo Sarat. un-ta, 2012. 192 p. (in Russian).
4. Bogomolov A. M., Salii V. N. *Algebraicheskie osnovy teorii diskretnykh sistem* [Algebraic foundations of the theory of discrete systems]. Moscow, Nauka, 1997. 368 p. (in Russian).
5. Abrosimov M. B. On the complexity of some problems related to graph extensions. *Math. Notes*, 2010, vol. 88, iss. 5, pp. 619–625. DOI: <https://doi.org/10.1134/S0001434610110015>



6. Abrosimov M. B. Minimal graph extensions. In: *Novye informatsionnye tekhnologii v issledovanii diskretnykh struktur* [New Information Technologies in the Study of Discrete Structures]. Tomsk, 2000, pp. 59–64 (in Russian).
7. Abrosimov M. B. *Minimal'nye rasshireniia 4-, 5-, 6- i 7-vershinnykh grafov* [Minimal extension of graphs with 4, 5, 6 and 7 vertices]. Saratov, Saratov State University, 2000, 26 p.; VINITI 06.09.2000, no. 2352-B00 (in Russian).
8. Brinkmann G. Isomorphism rejection in structure generation programs. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 2000, vol. 51, pp. 25–38. DOI: <https://doi.org/10.1090/dimacs/051/03>
9. McKay B. D., Piperno A. Practical Graph Isomorphism, II. *Journal of Symbolic Computation*, 2014, vol. 60, pp. 94–112. DOI: <https://doi.org/10.1016/j.jsc.2013.09.003>
10. Volga Regional Center for New Information Technologies. Available at: <http://prcnit.sgu.ru> (accessed 1 May 2019) (in Russian).
11. Graph World. Available at: <http://graphworld.ru> (accessed 1 May 2019) (in Russian).

Cite this article as:

Abrosimov M. B., Kamil I. A. K., Lobov A. A. Construction of All Nonisomorphic Minimal Vertex Extensions of the Graph by the Method of Canonical Representatives. *Izv. Saratov Univ. (N. S.), Ser. Math. Mech. Inform.*, 2019, vol. 19, iss. 4, pp. 479–486 (in Russian). DOI: <https://doi.org/10.18500/1816-9791-2019-19-4-479-486>
