УДК 519.178; 512.53 doi: 10.17238/issn2227-6572.2016.4.69

ПРИМЕНЕНИЕ АЛГОРИТМОВ ПРОВЕРКИ ИЗОМОРФИЗМА ГРАФОВ В ТЕОРИИ ПОЛУГРУПП

Л.В. Зяблицева, С.А. Пестов**

*Северный (Арктический) федеральный университет имени М.В. Ломоносова

Одной из наиболее интересных проблем теории полугрупп является проблема изоморфизма для данного класса полугрупп, состоящая в существовании алгоритма (отличающегося от алгоритма полного перебора), распознающего для любых двух полугрупп из данного класса, изоморфны они или нет. Аналогичная проблема есть и в теории графов, причем для некоторых классов графов этот вопрос решен. В статье рассмотрены полугруппы, являющиеся полурешетками, для проверки изоморфизма которых можно применить известные алгоритмы проверки изоморфизма графов. Описано, как для таких полугрупп можно найти соответствующий им граф. Этот граф может оказаться деревом, и в этом случае для проверки изоморфизма полугрупп можно применить известные алгоритмы проверки изоморфизма деревьев. Сформулирован и доказан критерий того, в каком случае граф полурешетки является деревом. Далее обосновывается выбор алгоритма проверки изоморфизма деревьев, описан этот алгоритм, представлена программа, написанная на языке *Haskell*, реализующая его. Чтобы применить выбранный алгоритм для проверки изоморфизма полурешеток, необходимо сначала полурешетке сопоставить дерево. Для этого авторами разработан и реализован также на языке *Haskell* необходимый алгоритм. Созданная в итоге программа для двух полурешеток, заданных таблицами Кэли, работает следующим образом: она выводит структуру соответствующих полурешеткам деревьев, каноническое имя полученных деревьев, проверяет изоморфизм деревьев, а значит, и полурешеток. При этом выбор и реализация алгоритмов являются эффективными, программа в течение нескольких секунд определяет изоморфизм полурешеток с трехзначным числом элементов.

Ключевые слова: *полугруппы; полурешетки; графы; деревья; изоморфизм полугрупп; изоморфизм графов; алгоритмы проверки изоморфизма полугрупп, графов, деревьев.*

Как в теории полугрупп, так и в теории графов проблема изоморфизма остается одной из наиболее интересных. Изоморфизм и инварианты в теории графов изучены более подробно, для графов частного типа созданы хорошие алгоритмы проверки изоморфизма. Один из самых известных и наиболее часто используемых видов графов – деревья. Для деревьев

Контактное лицо: Зяблицева Лариса Владимировна, *адрес:* 163002, г. Архангельск, наб. Северной Двины, д. 17; *e-mail:* zlarisav@yandex.ru.

Для цитирования: Зяблицева Л.В., Пестов С.А. Применение алгоритмов проверки изоморфизма графов в теории полугрупп // Вестн. Сев. (Арктич.) федер. ун-та. Сер.: Естеств. науки. 2016. № 4. С. 69–74. doi: 10.17238/issn2227-6572.2016.4.69.

разработаны различные алгоритмы проверки изоморфизма. Цель статьи – рассмотреть, как эти алгоритмы можно применить в теории полугрупп.

Графы полурешеток и частично упорядоченных множеств

Рассмотрим очень интересный класс коммутативных полугрупп, каждый элемент которых является идемпотентом (элемент x полугруппы S называется *идемпотентом*, если $x^2 = x$). Такие полугруппы называются *полурешетками.* Поясним, как можно построить граф для такой полугруппы [1, с. 10].

Пусть множество M частично упорядочено. Любое частично упорядоченное множество можно представить как ориентированный граф, в котором дуга (a, b) между парой элементов означает $a < b$ и нет такого элемента c, чтобы $a < c$ и $c < b$. Заметим, что не любой ориентированный граф является представлением частично упорядоченного множества. Для того чтобы ориентированный граф представлял частично упорядоченное множество, необходимо и достаточно, чтобы в нем не было циклов. В математической литературе частично упорядоченные множества обычно изображаются в виде неориентированных графов, при этом подразумевается, что предшествующие элементы расположены ниже (либо выше) последующих. Обозначим полученный в результате граф символом G_M.

Напомним, что частично упорядоченное множество называется *нижней полурешеткой,* если всякое двухэлементное его подмножество $\{a, b\}$ имеет точную нижнюю (обозначают $inf(a, b)$) грань. Двойственным образом дается определение верхней полурешетки (верхнюю грань двухэлементного подмножества обозначают $sup(a, b)$). Частично упорядоченное множество, являющееся и нижней, и верхней полурешеткой, называется *решеткой.*

Поясним, почему коммутативная полугруппа идемпотентов называется полурешеткой. Этот термин оправдан тем, что если рассмотреть на полурешетке S отношение естественного частичного порядка (заданного формулой $e \le f \Leftrightarrow e \cdot f = f \cdot e = e$), то для любых элементов e, f полурешетки S произведение $e \cdot f = inf(e, f)$; и обратно, если M – частично упорядоченное множество, в котором любые два элемента имеют точную нижнюю грань, то операция, заданная условием $a \cdot b = inf(a, b)$, превращает M в коммутативную полугруппу идемпотентов [2, с. 31].

Так как любой коммутативной полугруппе идемпотентов $(S, \cdot)$ соответствует частично упорядоченное множество $(S, \le)$, то любую такую полугруппу можно изобразить в виде графа. По построенному таким образом графу легко искать произведение элементов полурешетки: из вершин, соответствующих данным элементам, спускаемся вниз по ребрам до их пересечения. Элемент, стоящий на пересечении, является произведением исходных элементов.

Возникает вопрос: всегда ли является деревом полученный граф? Оказывается, нет. Дело в том, что для некоторых несравнимых элементов a и b коммутативной полугруппы идемпотентов $(S, \cdot)$ может существовать элемент c, отличный от элементов a и b, такой, что $c = sup(a, b)$. Тогда $c = sup(a, c) = sup(b, c)$. В этом случае в графе, соответствующем данной полугруппе, возникнет цикл.

Опишем коммутативные полугруппы идемпотентов $(S, \cdot)$, графы которых являются деревьями.

Пусть M – частично упорядоченное множество. *Нижним конусом* некоторого элемента a этого множества называется множество $L_a = \{x \in M \mid x \le a\}$. Частичный порядок $\le$ на множестве M называется *полулинейным снизу*, если для каждого элемента a этого множества его нижний конус линейно упорядочен.

Предложение. Представление коммутативной полугруппы идемпотентов $(S, \cdot)$ в виде графа G_S является деревом тогда и только тогда, когда частичный порядок, заданный формулой $e \le f \Leftrightarrow e \cdot f = e$, является полулинейным снизу.

Доказательство. Пусть для коммутативной полугруппы идемпотентов $(S, \cdot)$ ее граф G_S является деревом. Предположим, что порядок не является полулинейным снизу. Это означает,

что существует элемент a полугруппы, для которого его нижний конус не является линейно упорядоченным. Это, в свою очередь, означает, что существуют элементы из нижнего конуса этой полугруппы, которые не являются сравнимыми элементами. Пусть b и c – такие элементы. Но тогда имеем: $a = sup\ (a, c) = sup\ (a, b)$, поэтому в данном случае в графе G_S будет замкнутый маршрут $a \ldots b \ldots d \ldots c \ldots a$, где $d = inf(b, c)$ (на месте многоточий находятся вершины, располагающиеся в маршруте между соответствующими вершинами). Из этого маршрута можно выделить цикл, что противоречит определению дерева.

Пусть теперь частичный порядок в коммутативной полугруппе идемпотентов $(S, \cdot)$, заданный формулой $e \le f \Leftrightarrow e \cdot f = e$, является полулинейным снизу. Предположим, что соответствующий полугруппе граф G_S не является деревом. Это означает, что в нем есть цикл. Из алгоритма построения дерева следует, что для некоторых различных элементов полугруппы есть точные нижние и точные верхние грани и они отличны от самих элементов. Пусть b и c – такие элементы, $a = sup\ (b, c)$; $d = inf\ (b, c)$. Тогда элементы a, b, c, d принадлежат конусу $L_a = \{x \in S \mid x \le a\}$. Но тогда частичный порядок не является полулинейным снизу, так как нашлись элементы b и c конуса, которые не являются сравнимыми.

Из предложения следует, что проверку изоморфизма полурешеток, для которых частичный порядок, заданный формулой $e \le f \Leftrightarrow \Leftrightarrow e \cdot f = e$, является полулинейным снизу, можно произвести с помощью проверки изоморфизма соответствующих им деревьев.

Созданная в итоге программа работает следующим образом: по таблицам Кэли двух полурешеток, которые удовлетворяют предложению, строятся соответствующие им деревья, а затем проверяется изоморфизм построенных деревьев.

Алгоритм кодирования деревьев строками и его реализация

Для деревьев существуют различные эффективные алгоритмы проверки изоморфизма, в частности, их можно найти в [3, с. 64; 4; 5, с. 220; 6]. В [4] изоморфизм графов устанавливается с помощью их раскраски. В [5] описан алгоритм, с помощью которого любое дерево можно закодировать строкой из нулей и единиц, а затем восстановить дерево по найденной строке. Но при этом изоморфным деревьям могут быть сопоставлены различные строки. В [3] также описан алгоритм, при котором изоморфным деревьям могут быть сопоставлены различные строки. Для того чтобы соответствие между деревьями и строками было однозначным, можно применить алгоритм из [3], но несколько измененный. В алгоритме, описанном в [6], дополнительно производится лексикографическое упорядочение строк, за счет чего изоморфным деревьям сопоставляются одинаковые строки. Таким образом, задача проверки деревьев на изоморфизм сводится к задаче сравнения двух строк. Этот алгоритм прост в понимании, реализации и имеет небольшую сложность, поэтому и был нами выбран.

Рассмотрим *алгоритм кодирования деревьев строками.* Назовем его алгоритм A. Входные данные: дерево T и вершина r, являющаяся корнем дерева T. Результат выполнения алгоритма: строка. Обозначим результат выполнения алгоритма A для дерева T с корнем r так: $A\ (T, r)$. Шаги алгоритма:

1. Если дерево состоит всего из одной вершины, то сопоставим ему строку *«01»*.

2. Если в дереве более одной вершины, то рассмотрим вершины, являющиеся детьми корня дерева: $r_1, \ldots, r_n$. Для каждой из этих вершин построим деревья $T_1, \ldots, T_n$, имеющие корнями вершины $r_1, \ldots, r_n$.

3. Для каждого из полученных деревьев $T_1, \ldots, T_n$ запустим алгоритм A.

Замечание. В результате первых трех шагов всем концевым (висячим) вершинам дерева сопоставляется строка *«01»*.

4. Найдем вершины дерева, расстояние от которых до корня максимально и которым последовательность еще не сопоставлена.

5. Пусть u – такая вершина. Для вершины u найдем всех детей. Для этих детей ранее были

найдены соответствующие им последовательности S_1, …, S_l.

6. Строки S_1, …, S_l расположим в лексикографическом порядке (как слова в словаре). Получим упорядоченный набор строк s_1, …, s_l. Тогда вершине *u* сопоставим строку *«0s_1...s_l 1»*.

7. Если *u* – корень дерева *T*, то алгоритм остановим. Если нет, то перейдем к шагу 4.

Результатом работы алгоритма будет строка *«0s_1...s_n 1»*, где s_1, …, s_n – последовательности, которые сопоставлены детям корня *r* дерева *T*.

Для реализации алгоритма был выбран язык *Haskell*. На этом языке проще писать сложные программы, кроме того, программы, благодаря декларативности языка, получаются существенно короче, чем на других языках программирования. Также благодаря особенностям языка время работы программ минимальное.

Описанный алгоритм записи дерева строкой (каноническим именем) на языке *Haskell* выглядит следующим образом:

```
encodeTree :: Tree t -> String
encodeTree (Node root []) = «01»
encodeTree (Node root childs) = «0» +
+ intercalate «» encodeChilds + «1» where
encodeChilds = sort $ map encodeTree childs
```

Далее сравниваются канонические имена деревьев. Если они совпадают, то деревья изоморфны, иначе – нет:

```
isTreeIsomorphic :: Tree t1 -> Tree t2 -> Bool
isTreeIsomorphic t1 t2 = encodeTree t1 =
= encodeTree t2
```

Алгоритм перевода полурешетки в дерево и его реализация

Для того чтобы проверить, изоморфны ли две полугруппы, являющиеся полурешетками, необходимо сначала найти соответствующие им деревья. Этот алгоритм, как и алгоритм кодирования деревьев строками, является рекурсивным.

Деревья на языке *Haskell* представляются в виде: *Node rootNode forest*, где *rootNode* – корень дерева, а *forest* – список вершин, исходящих из корня. К примеру: *Node 0 [Node 1 [Node 3 []], Node 2 []]* – это дерево, состоящее из четырех вершин: *0, 1, 2, 3* – и трех ребер: *(0,1), (0,2), (1,3)*.

Рассмотрим *алгоритм перевода полурешетки в дерево*. Назовем этот алгоритм *semigroupToTree(m)*. Входные данные: полурешетка, заданная таблицей Кэли. Элементы таблицы: числа 0, 1, ..., *n* – 1, где *n* – порядок полурешетки. Результат выполнения алгоритма: дерево. Шаги алгоритма:

1. Найдем корень дерева *root*. Он равен произведению всех элементов полурешетки. Введем обозначение: $x = root$.

2. Найдем детей вершины *x*. Для этого переберем все элементы *y* полурешетки и отберем те из них, которые удовлетворяют условиям: $x \neq y$; $x \cdot y = x$; не существует такого элемента *z*, чтобы $z \neq x$; $z \neq y$; $z \cdot y = z$; $z \cdot x = x$. Собрав эти элементы в массив *rootChilds*, получим дерево *Node root rootChilds*.

3. Выполним шаг 2 для каждого элемента *x* массива *rootChilds* и их детей, пока не исчерпаем все элементы полурешетки.

Код на языке *Haskell*:

```
op m [x] = x
op m (x:y:xs) = op m $ (m !! x !! y):xs
am m = [0, 1..length m - 1]
f m x y z = and [z /= x, z /= y, op m [z, y] =
= z, op m [x, z] == x]
isNext m x y = and [x /= y, op m [x, y] =
= x, not $ any (f m x y) (am m)]
nexts m x = filter (isNext m x) (am m)
lowerElement m = op m $ am m
goStoT m x = [Node v (goStoT m v) | v <- nexts m x]
semigroupToTree m = Node le (goStoT m le) where le
= lowerElement m
```

Соберем все вместе и напишем функцию проверки полурешеток на изоморфизм:

```
isSemiIso s1 s2 = isTreeIsomorphic
(semigroupToTree s1) (semigroupToTree s2)
```

Пример применения программы

Приведем пример работы программы для двух полурешеток – S_1 и S_2, состоящих из 5 элементов, заданных с помощью таблиц Кэли

	0	1	2	3	4
0	0	0	0	0	0
1	0	1	0	0	0
2	0	0	2	2	2
3	0	0	2	3	2
4	0	0	2	2	4

	0	1	2	3	4
0	0	0	3	3	0
1	0	1	3	3	0
2	3	3	2	3	3
3	3	3	3	3	3
4	0	0	3	3	4

Рис 1. Таблицы Кэли для полурешеток S_1 и S_2

```
=== Tree of semigroup 1
===
0
|
+- 1
|
`- 2
   |
   +- 3
   |
   `- 4
Encode Tree: "0001011011"
```

```
=== Tree of semigroup 2
===
3
|
+- 0
|  |
|  +- 1
|  |
|  `- 4
|
`- 2
Encode Tree: "0001011011"
```

Рис. 2. Структура и канонические имена деревьев, соответствующих полурешеткам S_1 и S_2

(*рис. 1*). Для каждой полугруппы сначала выводится ее структура (как дерева), затем выводится ее каноническое имя (строка), затем канонические имена сравниваются и выдается ответ: *False* или *True*.

Для полурешеток, заданных данными таблицами, программа выдает результат, представленный на *рис. 2*.

Так как канонические имена деревьев совпадают, то делается вывод об изоморфизме деревьев, а значит, и полугрупп.

Для проверки времени работы данной программы было создана программа, генерирующая полурешетки по бинарным деревьям. Для двух таких полурешеток из 200 элементов программа работает менее 5 с.

Список литературы

1. *Зяблицева Л.В., Корабельщикова С.Ю., Попов И.Н.* Некоторые специальные полугруппы и их гомоморфизмы. Архангельск, 2013. 128 с.

2. *Артамонов В.А., Салий В.Н., Скорняков Л.А.* Общая алгебра. Т. 2. М., 1991. 480 с.

3. *Ахо А.В., Хопкрофт Д.Э., Ульман Д.Д.* Построение и анализ вычислительных алгоритмов. М., 1979. 521 с.

4. *Пономаренко И.Н.* Проблема изоморфизма графов: Алгоритмические аспекты. СПб., 2010. 57 с. URL: http://logic.pdmi.ras.ru/csclub/sites/default/files/graph_isomorphism_ponomarenko_lecture_notes.pdf (дата обращения: 06.12.2016).

5. *Гаврилов Г.П., Сапоженко А.А.* Задачи и упражнения по дискретной математике: учеб. пособие. М., 2006. 416 с.

6. *Smal A.* Explanation for «Tree Isomorphism" talk. Saint-Petersburg, 2008. 10 p. URL: http://logic.pdmi.ras.ru/~smal/files/smal_jass08.pdf (дата обращения: 06.12.2016).

References

1. Zyablitseva L.V., Korabel'shchikova S.Yu., Popov I.N. *Nekotorye spetsial'nye polugruppy i ikh gomomorfizmy* [Some Special Semigroups and Their Homomorphisms]. Arkhangelsk, 2013. 128 p.
2. Artamonov V.A., Saliy V.N., Skornyakov L.A. *Obshchaya algebra*. *T. 2*. [General Algebra. Vol. 2]. Moscow, 1991. 480 p.
3. Aho A.V., Hopcroft J.E., Ullman J.D. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.
4. Ponomarenko I.N. *Problema izomorfizma grafov: Algoritmicheskie aspekty* [The Problem of Graphs Isomorphism: Algorithmic Aspects]. Saint Petersburg, 2010. 57 p. Available at: logic.pdmi.ras.ru/csclub/sites/default/files/graph_isomorphism_ponomarenko_lecture_notes.pdf (accessed 06.12.2016).
5. Gavrilov G.P., Sapozhenko A.A. *Zadachi i uprazhneniya po diskretnoy matematike: ucheb. posobie* [Tasks and Exercises in Discrete Mathematics]. Moscow, 2006. 416 p.
6. Smal A. Explanation for "Tree Isomorphism" Talk. Saint Petersburg, 2008. 10 p. Available at: http://logic.pdmi.ras.ru/~smal/files/smal_jass08.pdf (accessed 06.12.2016).

doi: 10.17238/issn2227-6572.2016.4.69

Larisa V. Zyablitseva*, Sergey A. Pestov*

*Northern (Arctic) Federal University named after M.V. Lomonosov (Arkhangelsk, Russian Federation)

TEST ALGORITHMS OF THE GRAPH ISOMORPHISM IN THE THEORY OF SEMIGROUPS

One of the most interesting problems in the theory of semigroups is the isomorphism problem for this class of semigroups. This issue consists in the existence of an algorithm (which differs from the exhaustive algorithm) recognizing isomorphism of any two semigroups of a given class. A similar problem exists in the graph theory, and this issue has been resolved for some classes of graphs. The article considers semigroups, which are semilattices; their isomorphism can be checked by the known algorithms for the graph isomorphism testing. The corresponding graph can be found for these semigroups. This graph can be a tree; in this case we can apply the known algorithms of the trees isomorphism testing to verify the isomorphism of semigroups. The paper formulates and proves a criterion, when the semilattice graph is a tree. We have justified the choice of the algorithm of the trees isomorphism testing, described it, and presented a program written in Haskell that implements it. We should associate a tree with semilattice to apply the selected algorithm for the semilattice isomorphism testing. For this purpose the authors have developed and implemented an efficient algorithm in the Haskell language. The developed program for two semilattices, given by the Cayley tables, displays the structure of the trees relevant to semilattices, the canonical name of derived trees, tests the isomorphism of trees and semilattices. The choice and implementation of algorithms are effective; the program determines the semilattices isomorphism with a three-digit number of elements for a few seconds.

***Keywords**: semigroup; semilattice; graph; tree; isomorphism of semigroups; graph isomorphism; test algorithm of isomorphism of semigroups, graphs, trees.*

Received on May 30, 2016
Поступила 30.05.2016

***Corresponding author*:** Larisa Zyablitseva, *address*: Naberezhnaya Severnoy Dviny, 17, Arkhangelsk, 163002, Russian Federation; *e-mail*: zlarisav@yandex.ru.

For citation: Zyablitseva L.V., Pestov S.A. Test Algorithms of the Graph Isomorphism in the Theory of Semigroups. *Vestnik Severnogo (Arkticheskogo) federal'nogo universiteta. Ser.: Estestvennye nauki*, 2016, no. 4, pp. 69–74. doi: 10.17238/issn2227-6572.2016.4.69.