ЧЕБЫШЕВСКИЙ СБОРНИК
Том 22. Выпуск 2.

УДК 512.548.7 DOI 10.22405/2226-8383-2021-22-2-76-89

Об одном алгоритме проверки существования подквазигрупп[1]

А. В. Галатенко, А. Е. Панкратьев, В. М. Староверов

Галатенко Алексей Владимирович — кандидат физико-математических наук, Московский государственный университет имени М. В. Ломоносова (г. Москва).
e-mail: agalat@msu.ru
Панкратьев Антон Евгеньевич — кандидат физико-математических наук, Московский государственный университет имени М. В. Ломоносова (г. Москва).
e-mail: apankrat@intsys.msu.ru
Староверов Владимир Михайлович — кандидат физико-математических наук, Московский государственный университет имени М. В. Ломоносова (г. Москва).
e-mail: staroverovvl@imscs.msu.ru

Аннотация

Криптографические алгоритмы на основе квазигрупп активно изучаются в рамках перспективных исследований; кроме того, в последние годы регулярно появляются квазигрупповые алгоритмы-кандидаты на конкурсах криптографических стандартов. С точки зрения обеспечения стойкости одним из желательных требований, предъявляемых к квазигруппам, является отсутствие подквазигрупп (в противном случае преобразование может вырождаться). В работе предлагаются оптимизированные по временной сложности (за счет увеличения пространственной сложности) алгоритмы проверки наличия подквазигрупп и подквазигрупп порядка не меньше 2 в квазигруппах, заданных таблицей Кэли. Доказываются утверждения о сложности в худшем случае, а также приводятся оценки эффективности программной реализации на квазигруппах большого порядка. Результаты работы были анонсированы в рамках доклада на XVIII Международной конференции «Алгебра, теория чисел и дискретная геометрия: современные проблемы, приложения и проблемы истории».

Ключевые слова: квазигруппа, подквазигруппа, таблица Кэли.

Библиография: 16 названий.

Для цитирования:
А. В. Галатенко, А. Е. Панкратьев, В. М. Староверов. Об одном алгоритме проверки существования подквазигрупп // Чебышевский сборник, 2021, т. 22, вып. 2, с. 76–89.

[1]Работа выполнена при финансовой поддержке DRDO (Индия), проект “Quasigroup Based Cryptography: Security Analysis and Development of Crypto-Primitives and Algorithms (QGSEC)”, номер гранта SAG/4600/TCID/Prog/QGSEC”.

CHEBYSHEVSKII SBORNIK
Vol. 22. No. 2.

UDC 512.548.7 DOI 10.22405/2226-8383-2021-22-2-76-89

An algorithm for checking the existence of subquasigroups

A. V. Galatenko, A. E. Pankratiev, V. M. Staroverov

Galatenko Alexei Vladimirovich — candidate of physical and mathematical sciences, Lomonosov Moscow State University (Moscow).
e-mail: agalat@msu.ru
Pankratiev Anton Evgen'evich — candidate of physical and mathematical sciences, Lomonosov Moscow State University (Moscow).
e-mail: apankrat@shade.msu.ru
Staroverov Vladimir Mikhailovich — candidate of physical and mathematical sciences, Lomonosov Moscow State University (Moscow).
e-mail: staroverovvl@imscs.msu.ru

Abstract

Quasigroup-based cryptoalgorithms are being actively studied in the framework of theoretic projects; besides that, a number of quasigroup-based algorithms took part in NIST contests for selection of cryptographic standards. From the viewpoint of security it is highly desirable to use quasigroups without proper subquasigroups (otherwise transformations can degrade). We propose algorithms that take a quasigroup specified by the Cayley table as the input and decide whether there exist proper subquasigroups or subquasigroups of the order at least 2. Temporal complexity of the algorithms is optimized at the cost of increased spatial complexity. We prove bounds on time and memory and analyze the efficiency of software implementations applied to quasigroups of a large order. The results were reported at the XVIII International Conference «Algebra, Number Theory and Discrete Geometry: modern problems, applications and problems of history».

Keywords: quasigroup, subquasigroup, Cayley table.

Bibliography: 16 titles.

For citation:
A. V. Galatenko, A. E. Pankratiev, V. M. Staroverov, 2021, "An algorithm for checking the existence of subquasigroups", *Chebyshevskii sbornik*, vol. 22, no. 2, pp. 76–89.

Введение

В последние годы возник значительный интерес к использованию некоммутативных и неассоциативных алгебраических структур для построения криптографических примитивов (см., например, [1]). Примером таких структур являются конечные квазигруппы. Шеннон доказал, что основанное на квазигруппах табличное гаммирование обладает свойством совершенной секретности ([2]). Широкий спектр различных алгоритмов (хэш-функции, симметричные и асимметричные шифры, подписи) представлен в обзорах [3], [4]. В последних конкурсах NIST по выбору криптостандартов выступали и квазигрупповые кандидаты: хэш-функция EDON–R' ([5]), "легковесные" алгоритмы GAGE и InGAGE ([6]), а также модификация подписи MQQ–SIG ([7]).

Для обеспечения стойкости разумно выбирать квазигруппы, обладающие рядом дополнительных свойств. Например, полиномиальная полнота гарантирует NP-полноту задачи проверки разрешимости уравнений в соответствующей сигнатуре ([8]), а неаффинность — NP-полноту задачи проверки разрешимости систем уравнений ([9]). Известно, что почти все квазигруппы полиномиально полны и, следовательно, неаффинны (этот результат является простым следствием основной теоремы из работы [10]); таким образом, "случайные" квазигруппы оказываются хорошим выбором. Интересно, что сложность верификации полиномиальной полноты и неаффинности с помощью алгоритмов из [11], [12] оказывается сопоставимой со сложностью алгоритма порождения равномерного распределения на множестве квазигрупп заданного порядка из работы [13].

Еще одно желательное свойство — отсутствие подквазигрупп, так как в случае, если оба аргумента операции окажутся элементами подквазигруппы, результат также не выйдет за пределы подквазигруппы; это, в свою очередь, может упрощать задачу взлома за счет уменьшения пространства перебора. При этом в ряде случаев подквазигруппы порядка 1 считаются допустимыми. Так, при порождении квазигрупп с помощью правильных семейств функций над группами подквазигруппа порядка 1 существует и единственна ([14]), однако такие "неподвижные точки" могут быть учтены при построении криптоалгоритма.

В работе П. И. Собянина [15] был предложен кубический алгоритм для проверки существования подквазигрупп, описана программная реализация, распараллеленная для архитектуры CUDA, и приведены результаты работы программы на квазигруппах небольшого порядка (от 4 до 64). Идея алгоритма заключается в вычислении замыкания всевозможных одноэлементных подмножеств относительно квазигрупповой операции; таким образом, сложность в худшем случае составляет сложность вычисления одного замыкания умножить на число замыкаемых множеств. Несложно увидеть, что переход от одноэлементных подмножеств к двухэлементным даст решение задачи проверки наличия подквазигрупп порядка не меньше 2, при этом сложность увеличится на порядок (от кубической к полиному четвертой степени). Заметим, что, в силу жесткой ограниченности объема быстрой памяти у отдельных CUDA-вычислителей, неочевидно, что предложенное автором распараллеливание будет давать выигрыш в случае квазигрупп большого порядка.

В нашей работе описываются оптимизированные по времени алгоритмы для проверки существования произвольных собственных подквазигрупп и нетривиальных подквазигрупп (собственных подквазигрупп порядка не меньше 2). Временная сложность в первом случае составляет $O\left(n^{7/3} * (\log n)^{2/3}\right)$, во втором — $O\left(n^3 \log n\right)$. Основная идея оптимизации заключается в использовании монотонности замыкания множества элементов относительно операции квазигруппы: если множество A порождает множество B, то A содержится в нетривиальной подквазигруппе, только если B содержится в нетривиальной подквазигруппе. Таким образом, достаточно рассматривать только экстремальные относительно порождения начальные подмножества; эта идея будет подробно описана позднее. Практическая эффективность алгоритмов исследована на значительном количестве примеров квазигрупп большого порядка (вплоть до 2^{14}). Оказалось, что квазигруппы такого порядка могут быть обработаны за разумное время на обычной рабочей станции; дальнейшее увеличение порядка оказалось невозможным из-за исчерпания памяти. Результаты были анонсированы в рамках доклада на XVIII Международной конференции «Алгебра, теория чисел и дискретная геометрия: современные проблемы, приложения и проблемы истории» ([16]).

Дальнейшее изложение имеет следующую структуру. Во втором разделе вводятся необходимые определения. В третьем разделе формулируются и доказываются вспомогательные утверждения. Раздел 4 посвящен проверке существования подквазигрупп порядка не меньше 2, раздел 5 — проверке существования произвольных собственных подквазигрупп. В шестом разделе описывается программная реализация и приводятся результаты вычислительных

экспериментов. Наконец, раздел 7 представляет собой заключение.

Авторы выражают благодарность В. А. Артамонову за плодотворное обсуждение результатов и Д. В. Галатенко за изящную идею доказательства Леммы 1.

Основные определения и обозначения

Определение 1. *Конечной квазигруппой называется пара (Q, f), где Q — конечное множество, а операция $f\colon Q \times Q \to Q$ такова, что для любых $a, b \in Q$ уравнения $f(a, x) = b$ и $f(y, a) = b$ однозначно разрешимы.*

В дальнейшем все используемые структуры будут конечными, поэтому для краткости слово "конечный" будет опускаться.

Пусть $Q' \subseteq Q$. Обозначим через $f(Q')$ множество всех констант, получаемых при подстановке элементов Q' в формулы, порожденные f. Несложно увидеть, что такое замыкание обладает свойством монотонности: если $Q_1 \subseteq Q_2$, то $f(Q_1) \subseteq f(Q_2)$. Если $f(Q') = Q$, то множество Q' назовем полным. Очевидно, что для полноты Q' достаточно выполнения условия $|f(Q')| > \frac{|Q|}{2}$. В силу монотонности все надмножества полного множества также полны.

Определение 2. *Пусть $Q' \subset Q$, $1 \leqslant |Q'| < |Q|$. Если $f(Q') = Q'$, то говорим, что квазигруппа (Q, f) содержит собственную подквазигруппу Q' (точнее, подквазигруппу (Q', f'), где f' — ограничение операции f на $Q' \times Q'$). Если дополнительно выполнено условие $|Q'| \geqslant 2$, то подквазигруппа называется нетривиальной.*

Определение 3. *Пусть M, $|M| < \infty$ — некоторое множество, $k \in \mathbb{N}$, $M_1, \ldots, M_k$ — подмножества M. Множество $M_0 \subseteq M$ называется системой представителей для $M_1, \ldots, M_k$, если для всех i от 1 до k найдется элемент $m_i \in M_0$, такой что $m_i \in M_i$.*

Заметим, что представители для различных i могут совпадать.

Пусть $x \in \mathbb{R}$. Тогда $]x[$ означает верхнюю целую часть числа x, то есть минимальное целое число, большее или равное x.

В дальнейшем все логарифмы будут полагаться двоичными.

Вспомогательные утверждения

Лемма 1. *Пусть $|M| = n$, $M_1, \ldots, M_k \subset M$, $|M_1| = \ldots = |M_k| = t$. Тогда жадный алгоритм строит систему представителей M_0, такую что $|M_0| \leqslant \left(\frac{n}{t} + 1\right)] \log k + 2[$.*

Здесь под *жадным алгоритмом* подразумевается алгоритм, который стартует с пустого множества M_0, на каждом шаге добавляет в M_0 наиболее часто встречающийся среди $\{M_i\}$ элемент v и удаляет из набора можеств $\{M_i\}$ все M_i, содержащие v. Шаги алгоритма выполнятся, пока набор множеств $\{M_i\}$ непуст.

Доказательство. Рассмотрим первый этап алгоритма, в рамках которого элементы добавляются в множество M_0, пока количество оставшихся множеств M_i больше $k/2$. На каждом шаге первого этапа в наборе множеств $\{M_i\}$ суммарно содержится не менее $kt/2$ элементов, причем различных элементов не более n. Отсюда следует, что на каждом шаге наиболее часто встречающийся элемент среди набора множеств M_i встречается не менее, чем $\left]\frac{kt}{2n}\right[$ раз. Таким образом, за $\left]\frac{k/2}{]kt/(2n)[}\right[\leqslant \left]\frac{k/2}{kt/(2n)}\right[= \left]\frac{n}{t}\right[$ шагов будет удалено не менее $k/2$ множеств, и первый этап завершится.

Аналогично на втором этапе получаем, что не более, чем за $]n/t[$ шагов количество оставшихся множеств M_i станет не больше $k/4$. Продолжаем процесс, пока семейство множеств M_i содержит хотя бы один элемент. Несложно увидеть, что число этапов не превосходит $]\log k[+1 \leq]log[k[+2$, а общее количество шагов оценивается сверху величиной $\left(\frac{n}{t}+1\right)]\log k+2[$. □

Лемма 2. *Пусть (Q,f) — квазигруппа, $d \in \mathbb{N}$, $S_1,\ldots,S_k \subset Q$, $|S_1| = \ldots = |S_k| = d$, $M_i \subseteq f(S_i)$, $|M_i| \geqslant d$, $i = 1,\ldots,k$, M_i' — множество всех d-элементных подмножеств M_i, M_0 — система представителей для множеств M_i'. Тогда из полноты всех $m_0 \in M_0$ следует полнота всех S_i.*

Доказательство. Предположим противное: существует S_i, не являющееся полным. По условию найдется $m_0 \in M_0$, такое что $m_0 \in f(S_i)$. В силу монотонности замыкания

$$f(m_0) \subseteq f(f(S_i)) = f(S_i) \neq Q,$$

что противоречит условию. □

Лемма 2 допускает широкое обобщение (в частности, для случая несовпадающих мощностей и множеств M_i' более общего вида), однако для наших целей достаточно приведенной выше формулировки.

Лемма 3. *Пусть выполнены условия леммы 2 и дополнительно множества S_i представляют собой все d-элементные подмножества Q. Тогда квазигруппа (Q,f) имеет собственные подквазигруппы порядка $\geqslant d$ тогда и только тогда, когда существует $m_0 \in M_0$, не являющееся полным.*

Доказательство. Достаточность очевидна (искомая подквазигруппа есть $f(m_0)$). Необходимость вытекает из леммы 2 и того, что все d-элементные подмножества Q входят в $\{S_i\}$. □

Лемма 3 будет использована для обоснования корректности алгоритмов, представленных в разделах 4 (с $d=2$) и 5 (с $d=1$).

Алгоритм проверки существования нетривиальных подквазигрупп

Алгоритм из работы [15] осуществляет проверку существования собственных подквазигрупп, стартуя с одноэлементных подмножеств множества Q и вычисляя их замыкание. Однако для поиска нетривиальных подквазигрупп этого может оказаться недостаточно, как показывает следующий пример.

1	3	5	2	4
3	2	4	5	1
5	4	3	1	2
2	5	1	4	3
4	1	2	3	5

Таблица 1: Пример таблицы Кэли квазигруппы, в которой каждый элемент образует подквазигруппу порядка 1.

Нетрудно видеть, что каждое одноэлементное подмножество является подквазигруппой, и алгоритм закончит свою работу, не проверив существование нетривиальных подквазигрупп.

Для исправления ситуации достаточно стартовать с двухэлементных подмножеств множества Q. В дальнейшем при упоминании алгоритма из работы [15] мы будем иметь в виду и эту модификацию.

Алгоритм из [15] произведет следующие вычисления:
$(1,2) : 1, 2, 3(= 1 * 2), 5(= 1 * 3), 4(= 2 * 3)$
$(1,3) : 1, 3, 5(= 1 * 3), 4(= 1 * 5), 2(= 1 * 4)$
$(1,4) : 1, 4, 2(= 1 * 4), 3(= 1 * 2), 5(= 1 * 3)$
$(1,5) : 1, 5, 4(= 1 * 5), 2(= 1 * 4), 3(= 1 * 2)$
$(2,3) : 2, 3, 4(= 2 * 3), 5(= 2 * 4), 1(= 2 * 5)$
$(2,4) : 2, 4, 5(= 2 * 4), 1(= 2 * 5), 3(= 2 * 1)$
$(2,5) : 2, 5, 1(= 2 * 5), 3(= 2 * 1), 4(= 2 * 3)$
$(3,4) : 3, 4, 1(= 3 * 4), 5(= 3 * 1), 2(= 3 * 5)$
$(3,5) : 3, 5, 2(= 3 * 5), 4(= 3 * 2), 1(= 3 * 4)$
$(4,5) : 4, 5, 3(= 4 * 5), 1(= 4 * 3), 2(= 4 * 1)$

Так как каждое двухэлементное подмножество множества Q полно, нетривиальных подквазигрупп здесь нет.

Легко увидеть, что общая сложность алгоритма составляет $O\left(|Q|^4\right)$. Воспользуемся доказанными выше вспомогательными утверждениями и понизим сложность перебора — от множества всех пар перейдем к множеству пар существенно меньшего размера. Рассмотрим следующий алгоритм, состоящий из трех основных этапов:

1. рассматриваем всевозможные неупорядоченные пары $\{q_i, q_j \mid q_i, q_j \in Q\}$; вычисляем замыкание каждой пары, пока либо не получится замкнутое подмножество (в этом случае, очевидно, находится нетривиальная подквазигруппа и алгоритм завершает работу), либо размер частичного замыкания не станет равным $\left[c \cdot \sqrt{|Q|}\right]$ (положительная константа c является параметром);

2. каждое частичное замыкание рассматриваем как множество неупорядоченных пар входящих в него элементов и по лемме 1 строим систему представителей;

3. для каждой пары из системы представителей строим замыкание; по лемме 3 квазигруппа содержит нетривиальные подквазигруппы, если и только если найдется пара, замыкание которой не равно Q.

Реализация первого и третьего этапов алгоритма не вызывает проблем. Однако, первый этап алгоритма будет содержать некоторые дополнительные действия, необходимые для успешной реализации второй части алгоритма.

Будем с помощью $V_{i,j}$ обозначать частичное замыкание неупорядоченной пары $\{q_i, q_j\}$ (количество таких множеств равно $C_n^2 = n(n-1)/2$). Массивы $V_{i,j}$ должны строиться на первом этапе алгоритма. Параллельно с этим мы создадим криволинейный массив $t_{p,r}$, где t_p представляет собой массив пар $\{q_i, q_j\}$ таких, что $q_p \in V_{i,j}$. В реальной программе для того, чтобы сэкономить оперативную память, мы будем строить массивы $V_{i,j}$, но не будем их сохранять. При этом на первом этапе алгоритма нам придется строить массивы $V_{i,j}$ дважды: в процессе первого построения $V_{i,j}$ мы сможем определить размеры массивов t_p, после чего можно будет отвести под них память. В процессе второго построения $V_{i,j}$ мы сможем заполнить массивы t_p. Таким образом, указанные операции будут осуществлены за время $O\left(|Q|^3\right)$ при фиксированной константе c.

Заметим, что $V_{i,j}$ также можно рассматривать как множество всех неупорядоченных пар, образованных элементами частичного замыкания пары $\{q_i, q_j\}$. Из контекста будет ясно, какая интерпретация имеется в виду.

В процессе первого построения массивов $V_{i,j}$ мы также заполним двумерный массив G, где $G_{i,j}$ равно количеству частичных замыканий $V_{p,q}$, содержащих пару $\{q_i, q_j\}$. На это также потребуется $O\left(|Q|^3\right)$ времени при фиксированной константе c. Таким образом, первый этап алгоритма будет осуществлен за время $O\left(|Q|^3\right)$ при фиксированной константе c.

Здесь следует отметить, что в дальнейшем нам потребуется для заданной пары $\{q_a, q_b\}$ получать множество пар $\{q_i, q_j\}$, замыкания которых содержат $\{q_a, q_b\}$. Мы не можем себе позволить для каждой пары хранить множество пар $\{q_i, q_j\}\colon \{q_a, q_b\} \in V_{i,j}$ в силу ограничений на размер оперативной памяти. Вместо этого мы создали массив $t_{p,r}$, а пересечение массивов t_a и t_b как раз нам дает требуемое множество пар $\{q_i, q_j\}\colon \{q_a, q_b\} \subset V_{i,j}$.

Опишем детали реализации второго (наиболее сложного) этапа алгоритма. Второй этап алгоритма будет состоять из шагов, каждый из которых будет состоять из следующих пунктов:

1. За квадратичное время ищем максимум в массиве $G_{i,j}$; пусть максимум имеет индексы (a, b).

2. Ищем множество пар $\{q_i, q_j\}$, замыкания которых содержат $\{q_a, q_b\}$; это делается за квадратичное время с помощью нахождения пересечения массивов t_a и t_b .

3. Для всех $\{q_i, q_j\}$, найденных в предыдущем пункте, исключаем $V_{i,j}$ из семейства множеств частичных замыканий и, соответственно, корректируем значения массива G. Последнее делается вычитанием единицы изо всех $G_{k,l}$ таких, что $\{q_k, q_l\} \in V_{i,j}$. Легко увидеть, что суммарное время выполнения данного пункта во всем алгоритме $O\left(|Q|^3\right)$.

По лемме 1 (в которой в качестве n и k выступает мощность множества неупорядоченных пар, равная $\frac{|Q|(|Q|-1)}{2}$, а в качестве t выступает количество неупорядоченных пар в частичных замыканиях $V_{i,j}$, равное $\left\lceil c\sqrt{|Q|}\right\rceil \left(\left\lceil c\sqrt{|Q|}\right\rceil - 1\right)/2$), второй этап будет содержать $O\left(|Q|\log|Q|\right)$ шагов. Поэтому суммарное время работы второго пункта алгоритма во всех шагах равно $O\left(|Q|^3 \log|Q|\right)$. Отсюда мы сразу получаем требуемую оценку на количество шагов во второй и третьей частях алгоритма.

Осталось отметить, что при фиксированном значении параметра c алгоритм имеет пространственную сложность $O\left(n^2 \cdot \sqrt{|Q|}\right)$, поскольку общий объем массива t равен суммарному размеру массивов $V_{i,j}$, а массив G имеет квадратичный размер. При этом объем используемой памяти прямо пропорционален параметру c; таким образом, для работы с квазигруппами большого порядка значение c должно выбираться маленьким. С другой стороны, увеличение значения c снижает мощность системы представителей, таким образом понижая сложность самых сложных по времени шагов 2 и 3. Наконец, на третьем шаге потребуется вычислить $O\left(n\log|Q|\right)$ замыканий, причем каждое вычисление потребует времени $O\left(|Q|^2\right)$, и требуемая память пренебрежимо мала по сравнению с первым шагом. Таким образом, верно следующее утверждение.

Теорема 1. *При фиксированной константе c предложенный алгоритм устанавливает наличие нетривиальных подквазигрупп в квазигруппе порядка n с временной сложностью $O\left(n^3 \log n\right)$ и пространственной сложностью $O\left(c \cdot n^{5/2}\right)$, $n \to \infty$.*

В качестве примера, упрощающего понимание приведенного алгоритма, можно рассмотреть квазигруппу с таблицей Кэли, приведенной в Таблице 1. Пусть $c = 1.5$. В алгоритме будут строиться следующие частичные замыкания:
$(1,2) : 1, 2, 3 (= 1 * 2)$
$(1,3) : 1, 3, 5 (= 1 * 3)$
$(1,4) : 1, 4, 2 (= 1 * 4)$
$(1,5) : 1, 5, 4 (= 1 * 5)$

$(2,3) : 2,3,4(= 2 * 3)$
$(2,4) : 2,4,5(= 2 * 4)$
$(2,5) : 2,5,1(= 2 * 5)$
$(3,4) : 3,4,1(= 3 * 4)$
$(3,5) : 3,5,2(= 3 * 5)$
$(4,5) : 4,5,3(= 4 * 5)$

На первом этапе будет построен (вообще говоря, криволинейный) массив t (t_p = массив пар (i,j) таких, что $p \in V_{i,j}$):
$1 : \{1,2\},\{1,3\},\{1,4\},\{1,5\},\{2,5\},\{3,4\}$
$2 : \{1,2\},\{1,4\},\{2,3\},\{2,4\},\{2,5\},\{3,5\}$
$3 : \{1,2\},\{1,3\},\{2,3\},\{3,4\},\{3,5\},\{4,5\}$
$4 : \{1,4\},\{1,5\},\{2,3\},\{2,4\},\{3,4\},\{4,5\}$
$5 : \{1,3\},\{1,5\},\{2,4\},\{2,5\},\{3,5\},\{4,5\}$
и исходный массив G ($G_{i,j}$ равно количеству $V_{p,q}$, содержащих пару $\{i,j\}$):

	3	3	3	3
		3	3	3
			3	3
				3

Первый шаг второго этапа:
1) Находим элемент с максимальным значением $G_{i,j}$: $(1,2)$;
2) Ищем пересечение t_1 и t_2: $\{1,2\},\{1,4\},\{2,5\}$;
3) Удаляем из набора V: $V_{1,2}, V_{1,4}, V_{2,5}$;
Вычитаем по единице из всех $G_{i,j}$: $\{i,j\} \in V_{1,2}$ или $V_{1,4}$ или $V_{2,5}$.

Получаем G:

	0	2	2	2
		2	2	2
			3	3
				3

и оставшиеся V:
$(1,3) : 1,3,5(= 1 * 3)$
$(1,5) : 1,5,4(= 1 * 5)$
$(2,3) : 2,3,4(= 2 * 3)$
$(2,4) : 2,4,5(= 2 * 4)$
$(3,4) : 3,4,1(= 3 * 4)$
$(3,5) : 3,5,2(= 3 * 5)$
$(4,5) : 4,5,3(= 4 * 5)$

Второй шаг второго этапа:
1) Находим элемент с максимальным значением $G_{i,j}$: $(4,5)$;
2) Ищем пересечение t_4 и t_5: $\{1,5\},\{2,4\},\{4,5\}$;
3) Удаляем из набора V: $V_{1,5}, V_{2,4}, V_{4,5}$;
Вычитаем по единице из всех $G_{i,j}$: $\{i,j\} \in V_{1,5}$ или $V_{2,4}$ или $V_{4,5}$

Получаем G:

	0	2	1	1
		2	1	1
			2	2
				1

и оставшиеся V:
$(1,3) : 1,3,5(= 1 * 3)$
$(2,3) : 2,3,4(= 2 * 3)$
$(3,4) : 3,4,1(= 3 * 4)$
$(3,5) : 3,5,2(= 3 * 5)$

Третий шаг второго этапа:
1) Находим элемент с максимальным значением $G_{i,j}$: $(3,5)$;
2) Ищем пересечение t_3 и t_5: $\{1,3\},\{3,5\},\{4,5\}$;
3) Удаляем из набора V: $V_{1,3}, V_{3,5}, V_{4,5}$;
Вычитаем по единице из всех $G_{i,j}$: $\{i,j\} \in V_{1,3}$ или $V_{3,5}$ или $V_{4,5}$.

Оставшиеся V:
$(2,3) : 2,3,4(= 2 * 3)$
$(3,4) : 3,4,1(= 3 * 4)$

Четвертый шаг второго этапа:
$3 : \{1,2\},\{1,3\},\{2,3\},\{3,4\},\{3,5\},\{4,5\}$
$4 : \{1,4\},\{1,5\},\{2,3\},\{2,4\},\{3,4\},\{4,5\}$
1) Находим элемент с максимальным значением $G_{i,j}$: $(3,4)$;
2) Ищем пересечение t_3 и t_4: $\{2,3\},\{3,4\},\{4,5\}$;
3) Удаляем из набора V: $V_{2,3}, V_{3,4}, V_{4,5}$;
Вычитаем по единице из всех $G_{i,j}$: $\{i,j\} \in V_{2,3}$ или $V_{3,4}$ или $V_{4,5}$.

Набор V пуст.

Переходим к третьему этапу. На нем мы строим полные замыкания только для найденных пар (их существенно меньше, чем для алгоритма из [15]):
$(1,2) : 1,2,3(= 1 * 2), 5(= 1 * 3), 4(= 2 * 3)$
$(3,4) : 3,4,1(= 3 * 4), 5(= 3 * 1), 2(= 3 * 5)$
$(3,5) : 3,5,2(= 3 * 5), 4(= 3 * 2), 1(= 3 * 4)$
$(4,5) : 4,5,3(= 4 * 5), 1(= 4 * 3), 2(= 4 * 1)$

Поскольку все пары полны, делается вывод, что в данной квазигруппе нет нетривиальных подквазигрупп.

Описание алгоритма проверки существования собственных подквазигрупп

Не составляет труда применить описанный подход для проверки существования произвольных собственных подквазигрупп. В этом случае алгоритм будет также состоять из трех основных этапов:

1. рассматриваем всевозможные элементы $\{q_i \mid q_i \in Q\}$; вычисляем замыкание для каждого элемента (как одноэлементного подмножества), пока либо не получится замкнутое подмножество (в этом случае, очевидно, находится нетривиальная подквазигруппа и алгоритм завершает работу), либо размер частичного замыкания не станет равным $\left[c \cdot |Q|^{2/3} \cdot (\log |Q|)^{1/3}\right]$ (положительная константа c является параметром);

2. по лемме 1 строим систему представителей построенных множеств;

3. для каждого элемента из системы представителей (как одноэлементного подмножества) строим замыкание; в силу леммы 3 квазигруппа содержит собственные подквазигруппы, если и только если найдется представитель, замыкание которого не равно Q.

По аналогии с предыдущим алгоритмом будем с помощью V_i обозначать частичное замыкание элемента q_i (их число равно $|Q|$). Массивы V_i должны строиться на первом этапе алгоритма. Параллельно с этим мы создадим криволинейный массив $t_{p,r}$, где t_p представляет собой массив элементов q_i таких, что $q_p \in V_i$. В реальной программе мы также будем строить массивы V_i, но не будем их сохранять. При этом на первом этапе алгоритма нам придется строить массивы V_i дважды: в процессе первого построения V_i мы сможем определить размеры массивов t_p, после чего можно будет отвести под них память. В процессе второго построения V_i мы сможем заполнить массивы t_p. Таким образом, указанные операции будут осуществлены за время $O\left(|Q|^{7/3} \cdot (\log |Q|)^{2/3}\right)$ при фиксированной константе c: квадратичное от размера частичных замыканий время уйдет на обработку каждого из $|Q|$ замыканий.

В процессе первого построения массивов V_i мы также заполним массив G, где G_i равно количеству частичных замыканий V_p, содержащих элемент q_i.

Второй этап алгоритма будет сводиться к следующему: для каждого $q_i \in t_s$ (то есть элемента, замыкание которого содержит самый частый элемент q_s) исключаем V_i из семейства множеств частичных замыканий и, соответственно, корректируем значения массива G. Последнее делается вычитанием единицы из всех G_k таких, что $q_k \in V_i$, при этом корректируется значение минимума массива G (если уменьшаемый элемент массива G становится меньше минимума, то значение минимума и индекс минимального элемента изменяются). Легко увидеть, что суммарное время выполнения данного пункта во всем алгоритме по порядку совпадает с количеством элементов в массиве V, равным $O\left(|Q|^{5/3} \cdot (\log |Q|)^{1/3}\right)$.

Согласно лемме 1 (в которой в качестве n и k выступает мощность множества Q, а в качестве t выступает количество элементов в частичных замыканиях V_i, равное $\left[c \cdot |Q|^{2/3} \cdot (\log |Q|)^{1/3}\right]$), второй этап будет содержать $O\left(|Q|^{1/3} \cdot (\log |Q|)^{2/3}\right)$ шагов, но, как было указано ранее, суммарное время работы второго пункта алгоритма равно $O\left(|Q|^{5/3} \cdot (\log |Q|)^{1/3}\right)$. Отсюда мы сразу получаем требуемую оценку на количество шагов во второй и третьей частях алгоритма.

Осталось отметить, что при фиксированном значении параметра c алгоритм имеет пространственную сложность $O\left(|Q|^2\right)$. Действительно, квадратичный объем потребуется для хранения таблицы Кэли. Помимо этого, потребуется объем, равный $O\left(c \cdot |Q|^{5/3} \cdot (\log |Q|)^{1/3}\right)$, поскольку общий объем массива t равен суммарному размеру массивов V_i, а массив G имеет линейный размер. Заметим, что эта величина по порядку меньше, чем $|Q|^2$. На третьем шаге потребуется вычислить $O\left(|Q|^{1/3} \cdot (\log |Q|)^{2/3}\right)$ замыканий, причем каждое вычисление потребует времени $O\left(|Q|^2\right)$, и требуемая память пренебрежимо мала по сравнению с первым шагом. Таким образом, верно следующее утверждение.

Теорема 2. *При фиксированной константе c предложенный алгоритм устанавливает наличие собственных подквазигрупп в квазигруппе порядка n с временной сложностью $O\left(n^{7/3} * (\log n)^{2/3}\right)$ и пространственной сложностью $O\left(n^2\right)$, $n \to \infty$.*

Оценка практической эффективности

Для оценки практической эффективности алгоритма была создана программная реализация и проведен ряд вычислительных экспериментов на квазигруппах, использованных для тестов в работе [12].

Исходный код программной реализации вместе с файлом *Makiefile* и примером конфигурационного файла *config.txt* можно найти по ссылке *http://stargeo.ru/article20200808.zip* . В конфигурационном файле есть описания существенных параметров. Программа позволяет производить проверку наличия нетривиальных подквазигрупп алгоритмом из работы [15]

и алгоритмом, описанным в данной работе. Также есть возможность сравнения результатов работы этих алгоритмов.

С помощью данной программы произведено тестирование времени работы алгоритма поиска нетривиальных подквазигрупп из [15] и алгоритма, описанного в данной работе. Каждый алгоритм реализован с использованием и без использования OpenMP-распараллеливания. В следующей таблице приведены максимальные времена (в секундах) проверки наличия нетривиальных подквазигрупп (на 80 различных примерах, описанных в [12]) на 8-ядерной рабочей станции (i7–3770 CPU @3.40GHz) с 32 гигабайтами оперативной памяти:

N	Easy	EasyOMP	QC1	QC2	QC4	QC8	QC16	QP2	QP4	QP8
512	2	1	1							
1024	10	6	2							
2048	95	52	27	11	6	9	12	28	25	27
4096	1581	987	273	100	70	74	92	253	231	228
8192	17941	11426	2254	813	654	669	661	205	1 1979	1989
16384							5602			

Обозначения столбцов: N — порядок квазигруппы, Easy — алгоритм из работы [15]; EasyOMP — алгоритм из работы [15] с использованием OpenMP на 8 ядрах; QCV — быстрый алгоритм с параметром $c = 1/V$; QPN — быстрый алгоритм с использованием OpenMP на N ядрах, $c = 1$.

Следующая таблица является продолжением предыдущей таблицы для случаев использования OpenMP, $c < 1$:

N	QP2C2	QP4C2	QP8C2	QP2C4	QP4C4	QP8C24	QP4C16
2048	10	9	9	5	4	4	5
4096	93	81	82	49	32	32	35
8192	748	673	670	454	282	268	287
16384							2401

Обозначения столбцов: QPNCV — быстрый алгоритм с использованием OpenMP на N ядрах с параметром $c = 1/V$.

Проведенные расчеты показывают, что поставленную задачу можно успешно решать для квазигрупп порядков до 2^{14}. При этом задача допускает весьма эффективное распараллеливание. Использование коэффициента $c < 1$ позволяет существенно ускорить работу программы и уменьшить требуемый объем оперативной памяти. Так, расчеты для $N = 16384$ с помощью быстрого алгоритма из-за ограничений по памяти удалось провести только для $c = 1/16$. Алгоритм из работы [15] при таких порядках становится неприменимым из-за высокой сложности.

С помощью данной программы также произведено тестирование времени работы алгоритма проверки наличия собственных подквазигрупп из [15] и алгоритма, описанного в данной работе. Каждый алгоритм также реализован с использованием и без использования OpenMP-распараллеливания. В следующей таблице приведены максимальные времена (в секундах) проверки наличия собственных подквазигрупп на тех же примерах и на той же машине, что использовались в предыдущих численных экспериментах:

N	Easy	EasyOMP	QC0.25	QC0.5	QC1	QC2	QC4	QP4C0.25	QP4C0.5	QP4C1
4096	72	31								
8192	690	287								
16384	5639	2411	1	1	1	1	2	1	1	1
32762			3	2	3	4	6	2	1	1
65536			9	8	9	14	22	5	4	4

Обозначения столбцов аналогичны соответствующим обозначениям в предыдущих таблицах.

Заключение

В работе предложена оптимизация алгоритмов для проверки существования нетривиальных подквазигрупп и произвольных собственных подквазигрупп, а также приведена оценка практической эффективности этого алгоритма. В дальнейшем планируется рассмотрение других способов задания квазигрупп, прежде всего случай функционального задания.

СПИСОК ЦИТИРОВАННОЙ ЛИТЕРАТУРЫ

1. Markov V. T., Mikhalev A. V., Nechaev A. A. Nonassociative algebraic structures in cryptography and coding // Journal of Mathematical Sciences. 2020. Vol. 245, no. 2. P. 178–196. doi: 10.1007/s10958-020-04685-5

2. Shannon C. Communication theory of secrecy systems // Bell System Technical Journal. 1949. Vol. 28, no. 4. P. 656–715. doi: 10.1002/j.1538-7305.1949.tb00928.x

3. Глухов М. М. О применениях квазигрупп в криптографии // Прикладная дискретная математика. 2008. №2. С. 28–32.

4. Shcherbacov V. A. Quasigroups in cryptology // Computer Science Journal of Moldova. 2009. Vol. 17, no. 2(50). P. 193–228.

5. Gligoroski D., Ødegård R. S., Mihova M., Knapskog S. J., Drapal A., Klíma V., Amundsen J., El-Hadedy M. Cryptographic hash function EDON-R' // Proceedings of the 1st International Workshop on Security and Communication Networks. 2009. P. 1–9.

6. Gligoroski D. On the S-box in GAGE and InGAGE [электронный ресурс]. 2019. URL: `http://gageingage.org/upload/LWC2019NISTWorkshop.pdf` (дата обращения: 14.10.2020).

7. Gligoroski D., Ødegård R., Jensen R., Perret L., Faugère J.-C., Knapskog S., Markovski S. MQQ–SIG: an ultra-fast and provably CMA resistant digital signature scheme // INTRUST'11: Proceedings of the Third international conference on Trusted Systems. 2011. P. 184–203. doi: 10.1007/978-3-642-32298-3_13

8. Horváth G, Nehaniv Gh. L, Szabó Cs. An assertion concerning functionally complete algebras and NP-completeness // Theoretical Computer Science. 2008. Vol. 407. P. 591–595. doi: 10.1016/j.tcs.2008.08.028

9. Larose B., Zádori L. Taylor terms, constraint satisfaction and the complexity of polynomial equations over finite algebras // International Journal of Algebra and Computation. 2006. Vol. 16. P. 563–581. doi: 10.1142/S0218196706003116

10. Cameron P.J. Almost all quasigroups have rank 2 // Discrete Mathematics. 1992. Vol. 106–107. P. 111–115. doi: 10.1016/0012-365X(92)90537-P

11. Galatenko A. V., Pankratiev A.E. The complexity of checking the polynomial completeness of finite quasigroups // Discrete Mathematics and Applications. 2020. Vol. 30, no. 3. P. 169–175. doi: 10.1515/dma-2020-0016

12. Galatenko A. V., Pankratiev A.E., Staroverov V. M. Efficient verification of polynomial completeness of quasigroups // Lobachevskii Journal of Mathematics. 2020. Vol. 41, no. 8, accepted for publication.

13. Jacobson M. T., Matthews P. Generating uniformly distributed random latin squares // Journal of Combinatorial Designs. 1996. Vol. 4, no. 6. P. 405–437. doi: 10.1002/(SICI)1520-6610(1996)4:6<405::AID-JCD3>3.0.CO;2-J

14. Galatenko A. V., Nosov V. A., Pankratiev A.E. Latin squares over quasigroups // Lobachevskii Journal of Mathematics. 2020. Vol. 41, no. 2. P. 194–203. doi: 10.1134/S1995080220020079

15. Собянин П. И. Об алгоритме проверки наличия подквазигруппы в квазигруппе // Интеллектуальные системы. Теория и приложения. 2019. Т. 23, № 2. С. 79–84.

16. Галатенко А. В., Панкратьев А. Е., Староверов В. М. Об одном алгоритме проверки существования нетривиальных подквазигрупп // Материалы XVIII Международной конференции «Алгебра, теория чисел и дискретная геометрия: современные проблемы, приложения и проблемы истории», Тула, 2020, С. 150–153.

REFERENCES

1. Markov, V. T., Mikhalev, A. V. & Nechaev, A. A. 2020, “Nonassociative algebraic structures in cryptography and coding“, *J. Math. Sci.*, vol. 245, no. 2, pp. 178–196. doi: 10.1007/s10958-020-04685-5

2. Shannon, C. 1949, “Communication theory of secrecy systems“, *Bell Syst. tech.*, vol. 28, no. 4, pp. 656–715. doi: 10.1002/j.1538-7305.1949.tb00928.x

3. Glukhov, M. M. 2008, “Some applications of quasigroups in cryptography“, *Prikl. Discr. Mat.*, no. 2, pp. 28–32 (in Russian).

4. Shcherbacov, V. A. 2009, “Quasigroups in cryptology“, *CSJM*, vol. 17, no. 2(50), pp. 193–228.

5. Gligoroski, D., Ødegård, R. S., Mihova, M., Knapskog, S. J., Drapal, A., Klíma, V., Amundsen, J. & El-Hadedy, M. “Cryptographic hash function EDON-R“, *Proc. 1st Int. Wksh. on Security and Communication Networks*. Trondheim, 2009, pp. 1–9.

6. Gligoroski, D. On the S-box in GAGE and InGAGE (2019), Available at `http://gageingage.org/upload/LWC2019NISTWorkshop.pdf` (accessed 14 October 2020).

7. Gligoroski, D., Ødegård, R., Jensen, R., Perret, L., Faugère, J.-C., Knapskog, S. & Markovski, S. “MQQ–SIG: an ultra-fast and provably CMA resistant digital signature scheme“, *INTRUST’11: Proc. 3rd Int. Conf on Trusted Systems*. Beijing, 2011. pp. 184–203. doi: 10.1007/978-3-642-32298-3_13

8. Horváth, G, Nehaniv, Gh. L & Szabó, Cs. 2008, “An assertion concerning functionally complete algebras and NP-completeness“, *Theor. Comput. Sci.*, vol. 407, pp. 591–595. doi: 10.1016/j.tcs.2008.08.028

9. Larose, B. & Zádori, L. 2006, “Taylor terms, constraint satisfaction and the complexity of polynomial equations over finite algebras“, *Int. J. Algebra Comput.*, vol. 16, pp. 563–581. doi: 10.1142/S0218196706003116

10. Cameron, P.J. 1992, “Almost all quasigroups have rank 2“, *Discrete Math.*, vol. 116-117, pp. 111–115. doi: 10.1016/0012-365X(92)90537-P

11. Galatenko, A. V. & Pankratiev, A.E. 2020, “The complexity of checking the polynomial completeness of finite quasigroups“, *Discret. Math. Appl.*, vol. 30, no. 3, pp. 169–175. doi: 10.1515/dma-2020-0016

12. Galatenko, A. V., Pankratiev, A. E. & Staroverov, V. M. 2020, “Efficient verification of polynomial completeness of quasigroups“, *Lobachevskii J. Math.*, vol. 41, no. 8, pp. 1444–1453. doi: 10.1134/S1995080220080053.

13. Jacobson, M. T. & Matthews, P. 1996, “Generating uniformly distributed random latin squares“, *J. Comb. Des.*, vol. 4, no. 6, pp. 405–437. doi: 10.1002/(SICI)1520-6610(1996)4:6<405::AID-JCD3>3.0.CO;2-J

14. Galatenko, A. V., Nosov, V. A. & Pankratiev, A.E. 2020, “Latin squares over quasigroups“, *Lobachevskii J. Math.*, vol. 41, no. 2, pp. 194–203. doi: 10.1134/S1995080220020079

15. Sobyanin, P.I. 2019, “An algorithm that decides if a quasigroup contains subquasigroups“, *Intellektualnye sistemy. Teoria i prilojenia*, vol. 23, no. 2, pp. 79–84 (in Russian).

16. Galatenko, A. V., Pankratiev, A. E. & Staroverov, V. M. “An algorithm for checking the existence of nontrivial subquasigroups“, *Materialy XVIII Mejdunarodnoi Konferentsii “Algebra, Teoria Chisel i Discretnaya Geometria: Sovremennye Problemy, Prilojenia i Problemy Istorii“* (Proc. 18th Int. Conf. “Algebra, number theory and discrete geometry: modern problems, applications and problems of history“). Tula, 2020, pp. 150–153 (in Russian).