Данный текст является русскоязычной версией опубликованной на английском языке статьи и представлен в авторской редакции только на данном сайте!

UDC 519.6
Doi: 10.31772/2587-6066-2020-21-2-187-194

For citation**:** Kuznetsov A. A., Kishkan V. V. A routing algorithm for the Cayley graphs generated by permutation groups. *Siberian Journal of Science and Technology.* 2020, Vol. 21, No. 2, P. 187–194. Doi: 10.31772/2587-6066-2020-21-2-187-194.

Для цитирования: Кузнецов А. А., Кишкан В. В. Об одном алгоритме маршрутизации на графах Кэли, порожденных группами подстановок // Сибирский журнал науки и технологий. 2020. Т. 21, № 2. С. 187–194. DOI: 10.31772/2587-6066-2020-21-2-187-194.

ОБ ОДНОМ АЛГОРИТМЕ МАРШРУТИЗАЦИИ НА ГРАФАХ КЭЛИ, ПОРОЖДЕННЫХ ГРУППАМИ ПОДСТАНОВОК

А. А. Кузнецов*, В. В. Кишкан

Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
*E-mail: alex_kuznetsov80@mail.ru

Целью настоящей работы является создание эффективного алгоритма маршрутизации на графах Кэли групп подстановок, превосходящего по своим характеристикам алгоритм, использующий автоматическую структуру группы.

В первом разделе статьи описан вспомогательный алгоритм А–1, который позволяет нумеровать элементы заданной группы подстановок.

Во втором разделе представлен алгоритм А–2 для вычисления таблицы маршрутизации на графе Кэли и алгоритм А–3, который позволяет определить оптимальный маршрут между двумя произвольными вершинами графа. Для данных алгоритмов также получены оценки временной и пространственной сложности.

В третьем разделе описан алгоритм А–4, при помощи которого можно вычислить минимальное слово элемента группы. Доказано, что вычислительная сложность алгоритма будет пропорциональна длине входящего слова.

В четвертом разделе приведены результаты компьютерных экспериментов для некоторых групп подстановок, в которых сравнивается время вычисления минимальных слов по алгоритму А–4 и алгоритму, основанному на построении автоматической групповой структуры. Показано, что А–4 значительно быстрее своего конкурента.

Ключевые слова: граф Кэли, группа подстановок.

A ROUTING ALGORITHM FOR THE CAYLEY GRAPHS GENERATED BY PERMUTATION GROUPS

A. A. Kuznetsov*, V. V. Kishkan

Reshetnev Siberian State University of Science and Technology
31, Krasnoyarskii rabochii prospekt, Krasnoyarsk, 660037, Russian Federation
*E-mail: kuznetsov@sibsau.ru

The purpose of this work is to create an effective routing algorithm on the Cayley graphs of permutation groups, superior in its characteristics to an algorithm using an automatic group structure.

In the first section of the article we describe the auxiliary algorithm A–1 which allows numbering elements of a given permutation group.

In the second section we present the algorithm A–2 for calculating the routing table on the Cayley graph and algorithm A–3 for determination the optimal route between two arbitrary vertices of the graph. Estimates of time and space complexity are also obtained for these algorithms.

In the third section we describe the algorithm A–4 for calculation the minimal word of a group element. It is proved that the computational complexity of the algorithm will be proportional to the length of the input word.

The fourth section presents the results of computer experiments for some groups of permutation groups, which compare the time for calculating the minimum words using algorithm A – 4 and an algorithm based on the construction of an automatic group structure. It is shown that A – 4 is much faster than its competitor.

Keywords: Cayley graph, a permutation group.

Введение. В настоящее время постоянно увеличивающийся спрос на облачные вычисления приводит к росту крупномасштабных центров обработки данных (ЦОД).

Современные ЦОД содержат сотни тысяч узлов, соединенных между собой сетью. Топология такой сети, т. е. способ соединения узлов, является ключевым звеном, от которого зависит быстродействие, отказоустойчивость, надежность и другие характеристики ЦОД.

По этой причине проектирование сети является очень важной задачей, включающей в себя поиск моделей графов, которые имеют хорошие топологические свойства и позволяют использовать эффективные алгоритмы маршрутизации. Этими качествами обладают графы Кэли, имеющие такие привлекательные топологические свойства как высокая симметрия, иерархическая структура, рекурсивная конструкция, высокая связность и отказоустойчивость [1]. Определение графа Кэли подразумевает, что вершины графа являются элементами некоторой алгебраической группы. Выбор группы и ее порождающих элементов позволяет получить граф [2], отвечающий необходимым требованиям по диаметру, степени вершин, количеству узлов и т. д.

Пусть $G := \langle X \rangle$ – конечная группа, порожденная упорядоченным множеством $X := \{x_1 \prec x_2 \prec \ldots \prec x_m\}$, которое также называют алфавитом. Множество всех слов (строк) над алфавитом X будем обозначать X^*. Пусть $w := x_1 x_2 \ldots x_l$ – слово над X и $|w| := l$ – его длина. На множестве X^* также определим отношение порядка. Пусть v и w – два произвольных слова в алфавите X. Тогда $v \prec w$, если $|v| < |w|$, а в случае равенства длин слов, меньшее слово будет определяться согласно введенному лексикографическому порядку на порождающих. Если необходимо подчеркнуть, что строка $v \in X^*$ соответствует элементу $g \in G$, то мы будем писать v_g. Строку v будем называть минимальным словом элемента g, если для всех других $w \in X^*$, таких что $v_g = w_g$, будет выполняться $v \prec w$. Очевидно, что каждому $g \in G$ соответствует уникальное минимальное слово. Длиной элемента группы g будем называть длину его минимального слова v, т. е. $|g| := \min\{|v_g| : v_g \in X^*\}$. Заметим, что в общем случае задача по определению минимального слова является NP-сложной [3].

Графом Кэли $\Gamma := Cay(G, X)$ группы G относительно X называют ориентированный невзвешенный помеченный граф с множеством вершин $V(\Gamma) := \{g \mid g \in G\}$ и множеством

ребер $E(\Gamma) := \{(g, gx) | \ x \in X, g \in G\}$. Пусть $(g, gx) \in E(\Gamma)$, тогда генератор x называют меткой данного ребра. Если $X = X \cup X^{-1}$, то граф Г будет неориентированным. Будем считать, что единичный элемент $e \notin X$, т. е. в Г нет петель. Как известно [4], кратчайшее расстояние между двумя произвольными вершинами графа g и h, которое мы обозначим $d(g,h)$, равно длине минимального слова элемента $g^{-1}h$, т. е. $d(g,h) := | g^{-1}h |$.

Остановимся на известных к настоящему времени алгоритмах маршрутизации на графах Кэли. Традиционные методы, такие как алгоритмы Дейкстры или Беллмана – Форда, могут использоваться на графах любого вида, но требуют значительных пространственных и временных ресурсов [5]. Для некоторых семейств графов Кэли существуют особые алгоритмы маршрутизации, которые в отличие от традиционных методов, используют топологические характеристики графа, уменьшая при этом временную и/или пространственную сложность. Сюда можно отнести такие семейства графов, как гиперкуб [6], бабочка [7] и звездный граф [8], которые являются графами Кэли. В работе [9] представлен алгоритм маршрутизации для блинчиковых и звездных графов, основанный на сортировке перестановок. Однако этот подход не обеспечивает маршрутизацию по кратчайшему пути. К. Тэнг и Б. Арден доказывают [10], что все конечные графы Кэли могут быть представлены обобщенными хордальными кольцами, а затем предлагают итерационный алгоритм маршрутизации, основанный на таблице поиска. Пространственная сложность такого алгоритма составляет $O|G|^2)$, а временная – $O(D)$, где $|G|$ и D – размер и диаметр сети соответственно. Л. Вэнг и К. Тэнг для поиска кратчайших путей на графе Бореля предлагают алгоритм [11], который сначала вычисляет в автономном режиме таблицу маршрутизации от одного узла ко всем остальным; затем, используя свойство транзитивности графа Кэли, создает таблицу маршрутизации для всех узлов. Вычислительная сложность данного алгоритма ограничена $O(\log_4 |G|)$, а пространственная – $O(D \cdot |G|^2)$. В [12] представлен распределенный отказоустойчивый алгоритм маршрутизации на графе Бореля. Этот двухфазный алгоритм использует два типа таблиц маршрутизации: статическую и динамическую.

В статье [13] представлен алгоритм маршрутизации для специального класса графов Кэли, используемых в качестве топологии сети беспроводного ЦОД. Данный алгоритм является двухуровневым: для отправки сообщений между серверами в одной стойке и серверами в разных стойках. Каждый сервер идентифицируется тремя значениями: координаты стойки, ярус, на котором находится сервер, и его номер на ярусе. Кроме того, каждый сервер использует три таблицы маршрутизации для пересылки пакетов из источника в пункт назначения по кратчайшему маршруту.

Монография [14] представляет собой фундаментальную работу о взаимосвязи алгебраических групп и конечных автоматов. В этом случае на группе $G = \langle X \rangle$ определяется автоматная структура специального вида, используя которую можно вычислить минимальное слово для любого элемента группы.

Согласно [14], конечный автомат группы A считывает произвольное слово $w_g \in X^*$, обрабатывает его и выдает минимальное слово элемента g. При этом время T_0 обработки слова w будет пропорционально квадрату его длины, т. е. $T_0 = O(| w |^2)$.

Используя данный результат, в работе [15] был предложен алгоритм поиска кратчайшего пути на графе Кэли (обозначим его A–0), при этом его вычислительная сложность ограничена $O(D^2)$, а пространственная – $M_0 = O(| X | \cdot | G | + | \mathrm{A} |)$, где $| \mathrm{A} |$ – число состояний автомата группы.

Отметим, что все представленные выше алгоритмы маршрутизации могут быть отнесены в одну из следующих категорий: а) те, которые предназначены для конкретных графов Кэли;

б) универсальные с высокой пространственно-временной сложностью; в) с низкой сложностью, которые не обеспечивают кратчайших путей.

Целью настоящей работы является создание эффективного алгоритма маршрутизации на графах Кэли групп подстановок, превосходящего по своим характеристикам алгоритм из [15].

В первом разделе статьи описан вспомогательный алгоритм А–1, который позволяет нумеровать элементы заданной группы подстановок.

Во втором разделе представлен алгоритм А–2 для вычисления таблицы маршрутизации на графе Кэли. Далее описывается алгоритм А–3, который позволяет определить оптимальный маршрут между двумя произвольными вершинами графа. Для данных алгоритмов также получены оценки временной и пространственной сложности.

В третьем разделе описан алгоритм А–4, при помощи которого можно вычислить минимальное слово элемента группы. Доказано, что вычислительная сложность алгоритма будет пропорциональна длине входящего слова.

В четвертом разделе приведены результаты компьютерных экспериментов для некоторых групп подстановок, в которых сравнивается время вычисления минимальных слов по алгоритму А–4 и процедуре из [15].

В заключении рассматриваются перспективы развития проекта.

1. Алгоритмы на графах Кэли групп подстановок. Пусть G – конечная группа подстановок на множестве точек $\Omega = \{1, 2, \ldots, n\}$. Обозначим $\alpha^g := g[\alpha]$ образ элемента $\alpha \in \Omega$ под действием $g \in G$. Орбитой точки $\alpha \in \Omega$ называется множество $\alpha^G := \{\alpha^g \mid g \in G\}$. Стабилизатором точки $\alpha \in \Omega$ будем называть множество $G_\alpha := \{g \in G \mid \alpha^g = \alpha\}$. Для заданных $\beta_1, \beta_2, \ldots, \beta_i \in \Omega$ индуктивно определим

$$G_{\beta_1,\beta_2,\ldots,\beta_i} := (G_{\beta_1,\beta_2,\ldots,\beta_{i-1}})_{\beta_i} = \{g \in G \mid \beta_j^g = \beta_j,\ j = 0,1,\ldots,i\}.$$

Последовательность различных элементов $B := (\beta_1, \beta_2, \ldots, \beta_m)$ будем называть базой группы G, если $G_{\beta_1,\beta_2,\ldots,\beta_m} = e$. Таким, образом только единичный элемент группы оставляет неподвижными все точки базы. Пусть $G^{(i)} := G_{\beta_1,\beta_2,\ldots,\beta_{i-1}}$. Далее определим цепь стабилизаторов

$$G := G^{(1)} \geq G^{(2)} \geq \ldots \geq G^{(m)} \geq G^{(m+1)} = e.$$

Если $G^{(i+1)}$ является собственной подгруппой $G^{(i)}$ для $i \in [1, m]$, то базу B называют несократимой.

Смежные классы группы $G^{(i)}$ по подгруппе $G^{(i+1)}$ имеют взаимно однозначное соответствие с элементами орбиты $\Delta^{(i)} := \beta_i^{G^{(i)}}$. Действительно, если $a, b \in G^{(i)}$ и $G^{(i+1)}a = G^{(i+1)}b$, то для некоторого $h \in G^{(i+1)}$ будет выполняться $a = hb$, поэтому $\beta_i^a = \beta_i^{hb} = \beta_i^b$.

Указанный выше факт дает возможность вычислить семейство представителей смежных классов (трансверсаль) $U^{(i)}$ группы $G^{(i)} \bmod G^{(i+1)}$. Для каждого $\gamma \in \Delta^{(i)}$ определим $u_i(\gamma) \in G^{(i)}$, который отображает β_i в γ, т.е. $\beta_i^{u_i(\gamma)} = \gamma$. В частном случае, если $\beta_i \to \beta_i$, то $u_i(\beta_i) := e$.

Согласно введенным обозначениям, мы получим упорядоченную последовательность $U^{(i)} := (u_i(\gamma) \mid \gamma \in \Delta^{(i)})$. Очевидно, что $|U^{(i)}| = |\Delta^{(i)}|$.

Объединив все $U^{(i)}$, мы получим полное семейство представителей смежных классов группы:

$$U = \bigcup_{i=1}^{m} U^{(i)}.$$

По теореме Лагранжа $|G| = |G^{(1)} : G^{(2)}| \cdot |G^{(2)}| = |U^{(1)}| \cdot |G^{(2)}|$. Аналогично, $|G^{(2)}| = |G^{(2)} : G^{(3)}| \cdot |G^{(3)}| = |U^{(2)}| \cdot |G^{(3)}|$. Продолжив данный процесс, мы получим $|G| = |U^{(1)}| \cdot |U^{(2)}| \cdots |U^{(m)}|$.

Пусть $B = (\beta_1, \beta_2, \ldots, \beta_m)$ – база G, тогда для любого элемента группы мы можем определить его базовый образ $B^g := (\beta_1^g, \beta_2^g, \ldots, \beta_m^g)$.

Лемма 1. *Для $\forall g \in G$ базовый образ B^g имеет уникальное представление.*

Доказательство. Пусть $B^g = B^h$, тогда $B^{gh^{-1}} = B$. Поэтому, согласно определению базы, $gh^{-1} = e$. Следовательно, $g = h$. ■

Лемма 2. *Любой элемент $g \in G$ может однозначно быть записан в каноническом виде*

$$g := (u_m, u_{m-1}, \ldots, u_1) = u_m u_{m-1} \ldots u_1, \text{ где } u_i \in U^{(i)} \text{ и } i \in [1, m]. \quad (1)$$

Доказательство. Так как $g \in G$, то он содержится в некотором смежном классе группы G по подгруппе $G^{(2)}$, поэтому запишем его в виде $g = h_2 u_1$, где $h_2 \in G^{(2)}$ и $u_1 \in U^{(1)}$. Аналогично, $h_2 = h_3 u_2$, где $h_2 \in G^{(2)}$ и $u_2 \in U^{(2)}$. Продолжая данный процесс, получим искомую формулу. ■

В дальнейшем нам также понадобится множество обратных элементов полного семейства представителей смежных классов

$$\hat{U} = \bigcup_{i=1}^{m} \hat{U}^{(i)}, \text{ где } \hat{U}^{(i)} := (u_i^{-1}(\gamma) | \ \gamma \in \Delta^{(i)}).$$

Для быстрого поиска элементов в $U^{(i)}$ и $\hat{U}^{(i)}$ нам понадобится вспомогательный массив индексов $A_{m \times n}$, элементы которого определяются следующим образом:

$$A_{ij} := \begin{cases} a_{ij} \in [0, |U^{(i)}| - 1] \ - \text{ номер элемента } u_i(j) \text{ в } U^{(i)}, \text{ если } j \in \Delta^{(i)}; \\ -1 \text{ в противном случае.} \end{cases}$$

Ниже представлен алгоритм, который получив на входе слово, состоящее из произведения элементов $g_1 g_2 \ldots g_l$ группы G, ее базу B, полное семейство представителей смежных классов U и его инверсию $\hat{U}$, а также вспомогательный массив A, возвращает каноническое представление $u_m u_{m-1} \ldots u_1$ данного произведения.

Алгоритм А–1. $u_m u_{m-1} \ldots u_1 = \text{Factor}(g_1 g_2 \ldots g_l, U, \hat{U}, A, B)$

Вход: $g_1 g_2 \ldots g_l$, $\hat{U}$, A и B

Выход: $u_m u_{m-1} \ldots u_1$

1. $h := B$
2. **Для всех** $i = 1, 2, \ldots, l$
3. $\quad h := (h_1^{g_i}, h_2^{g_i}, \ldots, h_m^{g_i})$

4. **Для всех** $i = 1, 2, \ldots, m$

5. $j := h[i]$

6. $u_i := U^{(i)}[a_{ij}]$

7. $u_i^{-1} := \hat{U}^{(i)}[a_{ij}]$

8. $h := (h_1^{u_i^{-1}}, h_2^{u_i^{-1}}, \ldots, h_m^{u_i^{-1}})$

9. **Вернуть** $u_m u_{m-1} \ldots u_1$

Пусть $T_i = O(f)$ и $M_i = O(f)$ – верхние асимптотические оценки вычислительной и пространственной сложности i-го алгоритма соответственно.

Лемма 3. *Алгоритм А–1 корректен и* $T_1 = O(l \cdot m + m^2)$.

Доказательство. Корректность алгоритма следует из лемм 1 и 2. Сначала мы получаем базовый образ элемента $g := g_1 g_2 \ldots g_l$, а затем при каждом $i \in [1, m]$ стабилизируем точку $\beta_i \in B$. В итоге мы получим $B^{g u_1^{-1} \ldots u_m^{-1}} = B$. Следовательно, $g = u_m u_{m-1} \ldots u_1$.

Для оценки вычислительной сложности обратим внимание на то, что число операций в цикле 2–3 ограничено $O(l \cdot m)$, а в цикле 4–8 – $O(m^2)$, поэтому $T_1 = O(l \cdot m + m^2)$. ■

Разложение (1) дает возможность эффективно нумеровать все $g \in G$, используя метод перечисления элементов кортежа в смешанной системе счисления [16]. Пусть $c := (c_m, c_{m-1}, \ldots, c_1)$ – основание смешанной системы счисления, в которой $c_1 := 1$ и $c_i = c_{i-1} \cdot |U^{(i-1)}|$ для $i \geq 2$.

Пусть $g := u_m u_{m-1} \ldots u_1$. Определим биективное отображение

$$\mathrm{N} : u_m u_{m-1} \ldots u_1 \leftrightarrow (a_m, \ldots, a_1), \text{ где } a_i \in [0, |U^{(i)}| - 1] \text{ – номер элемента } u_i \text{ в } U^{(i)}.$$

Заметим, что вектор $(a_m, \ldots, a_1)$ представляет собой число $\mathrm{N}(g) \in [0, |G| - 1]$ в системе счисления со смешанным основанием $(c_m, \ldots, c_1)$.

Пусть $k := \mathrm{N}(g)$ – номер элемента g.

Лемма 4. *Вычислительная сложность* $\mathrm{N}(g)$ *и* $\mathrm{N}^{-1}(k)$ *не превышает* $O(m)$.

Доказательство. Пусть $g = u_m u_{m-1} \ldots u_1$ и a_i – номер u_i в $U^{(i)}$. Тогда, $k = \mathrm{N}(g) = a_m c_m + \ldots + a_1 c_1$.

В обратном случае $g = \mathrm{N}^{-1}(k) = u_m u_{m-1} \ldots u_1$, где факторы u_i вычисляются следующим образом:

1. **Для всех** $i = 1, 2, \ldots, m$

2. $a_i := k \mod |U^{(i)}|$

3. $k := \lfloor k / |U^{(i)}| \rfloor$

4. $u_i := U^{(i)}[a_i]$.

Очевидно, что число операций в данных процедурах не превышает $O(m)$. ■

Замечание 1. *Для определенности, будем полагать, что все последовательности* $U^{(i)}$ *начинаются с единичного элемента* e. *В этом случае* $\mathrm{N}(e) := 0$.

Отметим, что для нумерации элементов группы нам необходимо знать ее базу *B* и полное семейство представителей смежных классов *U*. Для их вычисления мы будем использовать известный алгоритм Шрайера – Симса, предложенный Ч. Симсом в 1970 г. [17]. В настоящее

время существует множество его модификаций [18]. Наиболее эффективные версии алгоритма имеют низкую вычислительную сложность и реализованы в таких системах компьютерной алгебры, как GAP, Magma и Mathematica, а также в библиотеке SymPy для языка Python.

2. Алгоритмы маршрутизации на графах Кэли. Для поиска кратчайших путей на графе Кэли $Cay(G, X)$ нам понадобится таблица маршрутизации $P_{2\times|G|}$, которую также называют родительским деревом [19]. Далее представлен алгоритм, который, получив на входе порождающее множество группы X, ее базу B, полное семейство представителей смежных классов U и его инверсию $\hat{U}$, а также вспомогательный массив A, возвращает указанную таблицу.

Здесь и далее нас будет интересовать случай $|X| \ll |G|$ и $n \ll |G|$.

Алгоритм А–2. $P = \mathrm{BFS}(X, U, \hat{U}A, B)$

Вход: X, U, $\hat{U}$, A и B

Выход: Таблица маршрутизации P на графе Кэли $Cay(G, X)$

1. $P_{2\times|G|} := [\infty \quad \dots \quad \infty]$
2. $k := \mathrm{N}(e) := 0$
3. $P[1][k] := -1$
4. $P[2][k] := -1$
5. $Q := \{k\}$
6. **Пока** $Q \neq \varnothing$
7. извлечь $q \in Q$ из очереди
8. **Для всех** $x \in X$
9. $s := \mathrm{Factor}(\mathrm{N}^{-1}(q)x, U, \hat{U}, A, B)$
10. $k := \mathrm{N}(s)$
11. **Если** $P[1][k] = \infty$
12. добавить k в очередь Q
13. $P[1][k] := q$
14. $P[2][k] := x$
15. **Вернуть** P

Теорема 1. *Алгоритм А–2 корректен и* $T_2 = O(m^2 \cdot |X| \cdot |G|)$.

Доказательство. Данный алгоритм представляет собой классический метод поиска в ширину на графе [19]. В этом случае вершина с номером единичного элемента e (согласно замечанию 1 имеем $\mathrm{N}(e) := 0$) будет являться корнем родительского дерева P. Пусть $gx = h$, где $\mathrm{N}(g) = k$ и $\mathrm{N}(h) = l$, тогда $P[1][l] := k$, $P[2][l] = x$. Это означает, что вершина k является родителем вершины l и x – метка ребра (k, l).

Нам необходимо проверить всего $|X| \cdot |G|$ элементов. Время выполнения проверки каждого элемента, согласно леммам 3 и 4, ограничено $O(m^2)$. Следовательно, $T_2 = O(m^2 \cdot |X| \cdot |G|)$.

Следующий алгоритм при помощи таблицы маршрутизации P вычисляет кратчайший путь $w := x_1 x_2 \dots x_s$ между вершинами $a, b \in Cay(G, X)$, где $x_i \in X$. Очевидно, что $s \leq D$, где D – диаметр графа.

Алгоритм А–3. $w = \mathrm{Route}(a, b, U, \hat{U}A, B, P)$

Вход: a, b, U, $\hat{U}$, A, B и P

Выход: $w := x_1 x_2 \ldots x_s$ – кратчайший маршрут из вершины a в вершину b

1. $w := [\,]$ – пустое слово
2. $g_1 := \mathrm{N}^{-1}(a)$
3. $g_2 := \mathrm{N}^{-1}(b)$
4. $g := \mathrm{Factor}(g_1^{-1} g_2, U, \hat{U}, A, B)$
5. $k := \mathrm{N}(g)$
6. $l := k$
7. **Пока** $P[1][k] \neq -1$
8. $\quad k := P[1][l]$
9. $\quad w := P[2][l] \oplus w$ – конкатенация строк
10. $\quad l := k$
11. **Вернуть** w

Теорема 2. *Алгоритм А–3 корректен,* $T_3 = O(m^2 + D)$ *и* $M_3 = O(m \cdot n + |G|)$.

Доказательство. В графе Кэли кратчайший путь от вершины a к вершине b будет представлять собой минимальное слово $w := x_1 x_2 \ldots x_s$ элемента $g := g_1^{-1} g_2$, где $g_1 := \mathrm{N}^{-1}(a)$, $g_2 := \mathrm{N}^{-1}(b)$ и $x_i \in X$ [4]. Пусть k – номер вершины элемента g. Двигаясь по родительскому дереву P от вершины k к его корню, мы получим искомый путь.

Время выполнения пунктов 1–6 ограничено $O(m^2)$, а цикла 7–10 – $O(D)$. В итоге получим $T_3 = O(m^2 + D)$.

Пространственная сложность переменных U, $\hat{U}$, A и B ограничена $O(m \cdot n)$, а таблицы P – $O(|G|)$. Поэтому, $M_3 = O(m \cdot n + |G|)$. ■

Замечание 2. *Во многих прикладных задачах при исследовании графа Кэли* $\Gamma := Cay(G, X)$ *порядок порождающей группы* G *значительно превышает ее степень, т. е.* $m \leq n \ll |G|$. *В этом случае* $M_3 = O(|G|)$.

Пример. Рассмотрим группу $G = \langle X \rangle$, порожденную двумя циклами $x = (1,5,4)$ и $y = (3,4)$. Вычислим в GAP базу G и полное семейство представителей смежных классов:

$$B = (1,3,4),$$

$$U^{(1)} = (e, (1,4)(3,5), (1,5)(3,4), (1,3)(4,5)),\ \Delta^{(1)} = (1,4,5,3),$$

$$U^{(2)} = (e, (3,4,5), (3,5,4)),\ \Delta^{(2)} = (3,4,5),$$

$$U^{(3)} = (e, (4,5)),\ \Delta^{(3)} = (4,5).$$

При помощи Алгоритма 2 получим граф Кэли $\Gamma = Cay(G, X)$ и его родительское дерево P (рис. 1).

Для иллюстрации Алгоритма 3 найдем расстояние от вершины $a := 15$ до $b := 22$.

$$g_1 := \mathrm{N}^{-1}(a) = U^{(3)}[1] U^{(2)}[0] U^{(1)}[3],$$

$$g_2 := \mathrm{N}^{-1}(b) = U^{(3)}[1] U^{(2)}[2] U^{(1)}[2],$$

$$g := \mathrm{Factor}(g_1^{-1} g_2) = U^{(3)}[0] U^{(2)}[2] U^{(1)}[0],$$

$$k := \mathrm{N}(g) = 8.$$

Двигаясь по родительскому дереву P от вершины с номером 8 к его корню, получим

$$8 \xrightarrow{y} 20 \xrightarrow{x} 17 \xrightarrow{x} 14 \xrightarrow{y} 6 \xrightarrow{x} 0.$$

Следовательно, $w := xyxxy$. На графе Γ маршрут будет иметь следующий вид:

$$15 \xrightarrow{x} 19 \xrightarrow{y} 1 \xrightarrow{x} 4 \xrightarrow{x} 10 \xrightarrow{y} 22.$$

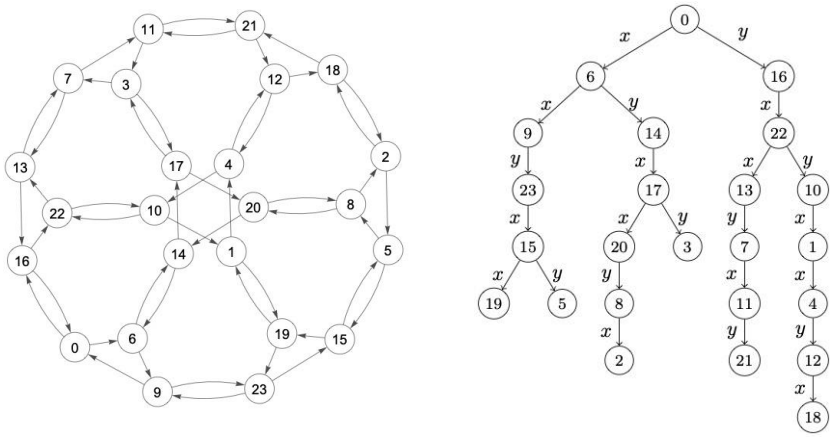


Fig. 1. The Cayley graph Γ = *Cay*(*G*, *X*) and its parent tree.
The edges of the graph Γ with the label *x* are represented as a straight line and *y* as an arc

Рис. 1. Граф Кэли Γ = *Cay*(*G*, *X*) и его родительское дерево *P*.
Ребра графа Γ с меткой *x* представлены в виде прямой линии, а *y* в форме дуги

3. Проблема минимального слова. Небольшая модификация алгоритма A–3 позволяет вычислить минимальное слово w из произвольной строки $v := x_1 x_2 \ldots x_r$ в алфавите порождающих X.

Алгоритм A–4. $w = \text{MinimalWord}(v, U, \hat{U} A, B, P)$

Вход: v, U, $\hat{U}$, A, B и P

Выход: $w := x_1 x_2 \ldots x_s$ – минимальное слово

1. $w := [\,]$ – пустое слово
2. $g := \text{Factor}(v, U, \hat{U}, A, B)$
3. $k := \text{N}\ (g)$
4. $l := k$
5. **Пока** $P[1][k] \neq -1$
6. $\quad k := P[1][l]$
7. $\quad w := P[2][l] \oplus w$ – конкатенация строк
8. $\quad l := k$
9. **Вернуть** w

Теорема 3. $T_4 = O(|v|)$ и $M_4 = O(m \cdot n + |G|)$.

Доказательство. Согласно леммам 3 и 4, время выполнения пунктов 1–4 ограничено $O(|v| \cdot m + m^2)$. Пункты 5–9 алгоритма представляют собой восходящее движение по родительскому дереву P, поэтому сложность этого участка не превышает $O(D)$. Таким образом, мы получим $T_4 = O(|v| \cdot m + m^2 + D)$. Если длина слова велика, то $T_4 = O(|v|)$.

Пространственная сложность переменных U, $\hat{U}$, A и B ограничена $O(m \cdot n)$, а таблицы P – $O(|G|)$, поэтому $M_4 = O(m \cdot n + |G|)$. ■

4. Сравнительный анализ алгоритмов. В следующей таблице приведены оценки временной и пространственной сложности алгоритмов A–0 и A–4.

Алгоритм	T_i	M_i
A–0	$O(\|v\|^2)$	$O(\|X\|\cdot\|G\|+\|A\|)$
A–4	$O(\|v\|)$	$O(m\cdot n+\|G\|)$

Из таблицы видно, что алгоритм A–4 имеет более низкую вычислительную сложность в сравнении с алгоритмом A–0.

Что касается пространственной сложности, то согласно замечанию 2, для многих интересных топологий будет выполняться $M_0 \sim M_4 \sim |G|$.

A–4 был реализован автором на языке C++, в свою очередь A–0 написан на языке C и является частью свободно распространяемого пакета KBMAG. Для того, чтобы обеспечить чистоту эксперимента, данные алгоритмы транслировались компилятором GCC с одинаковыми параметрами. На начальном этапе для тестирования были взяты две группы:

а) симметрическая группа $S_9 = \langle Y \rangle$, где $Y = \{(i, i+1) | \ i \in [1,8]\}$. Граф Кэли $Cay(S_9, Y)$ называют также графом пузырьковой сортировки. Хорошо известно, что $S_n = n!$, $m = n-1$ и $D_Y(S_n) = \dfrac{n(n-1)}{2}$;

б) спорадическая простая группа Мэтьё $M_{22} = \langle X \rangle$, где $X = \{x_1, x_2, x_2^{-1}\}$ и

$x_1 = (1,13)(2,8)(3,16)(4,12)(6,22)(7,17)(9,10)(11,14)$;

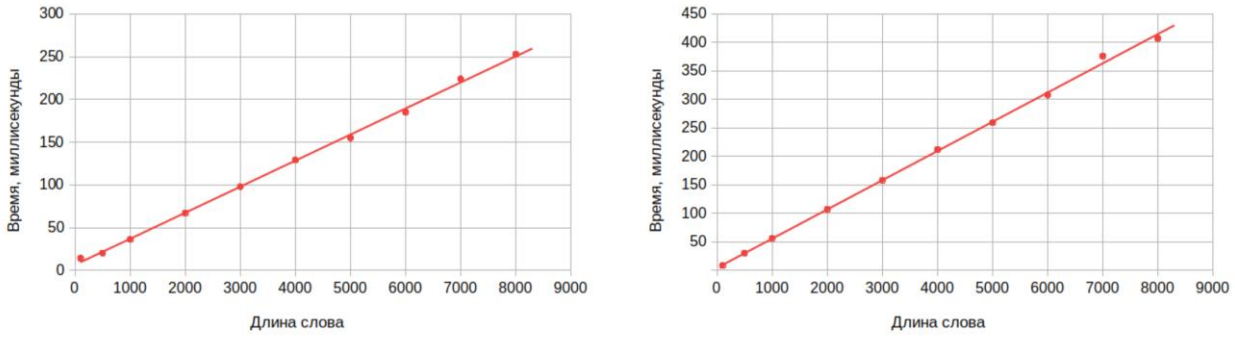
$x_2 = (1,22,3,21)(2,18,4,13)(5,12)(6,11,7,15)(8,14,20,10)(17,19)$.

При этом $|M_{22}| = 443520$, $m = 5$ и $D_X(M_{22}) = 34$.

На рис. 2 приведены графики зависимости времени $T_4(l)$ выполнения алгоритма A–4 для групп $S_9 = \langle Y \rangle$ и $M_{22} = \langle X \rangle$ в зависимости от длины входящего слова.

На рис. 3 приведены графики зависимости времени $T_0(l)$ и $T_4(l)$ от длины входящего слова для указанных групп.

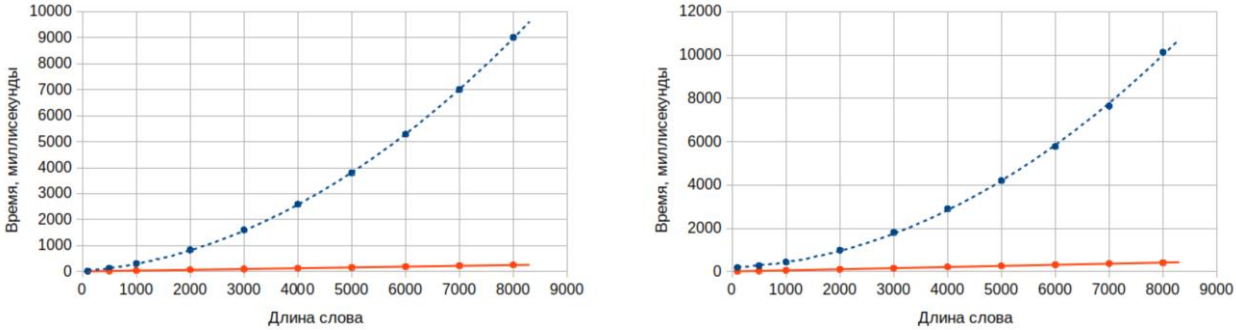
Эти графики наглядно показывают, что алгоритм A–4 значительно быстрее, чем A–0.



S_9 M_{22}

Fig. 2. Graphs of $T_4(l)$ for $S_9 = \langle Y \rangle$ and $M_{22} = \langle X \rangle$

Рис. 2. Графики $T_4(l)$ для групп $S_9 = \langle Y \rangle$ и $M_{22} = \langle X \rangle$



S_9 M_{22}

Fig. 3. Graphs of $T_0(l)$ (dotted line) and $T_4(l)$ for groups $S_9 = \langle Y \rangle$ and $M_{22} = \langle X \rangle$

Рис. 3. Графики $T_0(l)$ (пунктир) и $T_4(l)$ для групп $S_9 = \langle Y \rangle$ и $M_{22} = \langle X \rangle$

Заключение. Представленные в данной работе алгоритмы послужат отправной точкой для создания новых ресурсно-эффективных алгоритмов маршрутизации. Здесь можно выделить два направления. Во-первых, создание алгоритмов, учитывающих топологию сети. В этом случае алгоритмы будут проектироваться для конкретных классов графов Кэли. Во-вторых, разработка гибридных алгоритмов, включающих в себя как статическую, так и динамическую таблицы маршрутизации. Это позволит рассчитывать оптимальные маршруты в зависимости от текущего состояния сети.

References

1. Heydemann M. Cayley graphs and interconnection networks, in Graph symmetry: algebraic methods and applications (Editors: Hahnand Sabidussi). *Dordrecht: Kluwer Academic Publishers*. 1997, P. 167–226.

2. Loz E. New record graphs in the degree-diameter problem. *Australasian Journal of Combinatorics*. 2008, Vol. 41, P. 63–80.

3. Even S., Goldreich O. The Minimum Length Generator Sequence is NP–Hard. *Journal of Algorithms*. 1981, Vol. 2, P. 311–313.

4. Holt D., Eick B., O'Brien E. *Handbook of computational group theory*. Boca Raton: Chapman & Hall/CRC Press, 2005, 514 p.

5. Schibell S., Stafford R. Processor interconnection networks and Cayley graphs. *Discrete Applied Mathematics*. 1992, Vol. 40, P. 337–357.

6. Stamoulis G., Tsitsiklis J. Effcient routing Scheme for Multiple Broadcasts in Hypercubes. *IEEE Trans. on Parallel and Distributed Systems.* 1993, Vol. 4(7), P. 725–739.

7. Stamoulis G., Tsitsiklis J. The Effciency of Greedy Routing in Hypercubes and Butteries. *IEEE Transaction on Communication*. 1994, Vol. 42(11), P. 3051–3061.

8. Kiasari A., Sarbazi-Azad H. Analytic performance comparison of hypercubes and star graphs with implementation constraints. *Journal of Computer and System Sciences*. 2008, No. 6, P. 1000–1012.

9. Akers S., Krishnamurthy B. A group theoretic model for symmetric interconnection networks. *Proceedings of the International Conference on Parallel Processing*. 1986, P. 216–223.

10. Tang K., Arden B. Vertex-transitivity and routing for Cayley graphs in GCR representations. *Proceedings of ACM Symposium on Applied Computing SAC*. 1992, P. 1180–1187.

11. Wang L., Tang K. Topology-Based Routing for Xmesh in Wireless Sensor Networks. *Lecture Notes in Electrical Engineering*. 2009, Vol. 44, P. 229–239.

12. Ryu J., Noel E., and Tang K. Fault-tolerant Routing on Borel Cayley Graph. *IEEE ICC Next Generation Networking Symposium*. 2012, P. 2872–2877.

13. Shin J., Sirer E., Weatherspoon H., Kirovski D. On the feasibility of completely wireless datacenters. *EEE/ACM Transaction On Networking*. 2013, Vol. 21(5), P. 1666–1679.

14. Epstein D., Paterson M., Cannon J., Holt D., Levy S. and Thurston W. *Word Processing in Groups*. Boston: Jones and Barlett Publishers, 1992, 330 p.

15. Camelo M., Papadimitriou D., Fabrega L., Vila P. Efficient Routing in Data Center with Underlying Cayley Graph. *Proceedings of the 5th Workshop on Complex Networks CompleNet*. 2014, P. 189–197.

16. Knuth D. *The Art of Computer Programming, Volume 4A: Combinatorial Algorithms, Part 1*. Boston: Addison-Wesley Professional, 2011, 912 p.

17. Sims C. Computational methods in the study of permutation groups, *Computational problems in abstract algebra (Pergamon Press, Oxford)*. 1970, P.169–183.

18. Seress A. *Permutation Group Algorithms*. Cambridge: Cambridge University Press, 2003, 274 p.

19. Skiena S. *The Algorithm Design Manual*. London: Springer Science+Business Media, 2008, 730 p.

Кузнецов Александр Алексеевич – доктор физико–математических наук, профессор, директор НОЦ «Институт космических исследований и высоких технологий»; Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева. E-mail: alex_kuznetsov80@mail.ru.

Кишкан Владимир Владимирович – аспирант; Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева. E-mail: kishkan@mail.ru.

Kuznetsov Alexander Alexeevich – D. Sc., Professor, Head of Institute of Space Research and High Technologies; Reshetnev Siberian State University of Science and Technology. E-mail: alex_kuznetsov80@mail.ru.

Kishkan Vladimir Vladimirovich – Postgraduate student; Reshetnev Siberian State University of Science and Technology. E-mail: kishkan@mail.ru.