Записки научных
семинаров ПОМИ
Том 279, 2001 г.

А. В. Малютин

БЫСТРЫЕ АЛГОРИТМЫ РАСПОЗНАВАНИЯ И СРАВНЕНИЯ КОС

§0. Введение

В настоящей работе мы занимаемся вычислительной сложностью алгоритмов, разрешающих проблемы тождества и сравнения (для известных и новых порядков) в группах классов отображений поверхностей и, в частности, в группе кос.

0.1. Основные понятия, термины и обозначения. Пусть S – компактная связная поверхность (возможно, с непустым краем и/или ориентированная) и $P_{\mathrm{in}} \subset \operatorname{int} S$ – конечное множество отмеченных (выколотых) внутренних точек. Поверхность $S \setminus P_{\mathrm{in}}$ обозначим $S_{(\cdot)}$ (желая указать количество выколотых точек, будем писать также $S_{(n)}$ при $|P_{\mathrm{in}}| = n$). *Группой классов отображений* поверхности $S_{(\cdot)}$ (mapping class group – $\mathrm{MCG}(S_{(\cdot)})$) называют группу $\mathcal{H}(S_{(\cdot)}, \partial S)/\mathcal{H}_0(S_{(\cdot)}, \partial S)$, где $\mathcal{H}(S_{(\cdot)}, \partial S)$ – группа гомеоморфизмов $S_{(\cdot)}$, сохраняющих ориентацию (если она есть) и границу поточечно, $\mathcal{H}_0(S_{(\cdot)}, \partial S)$ – гомеоморфизмы, изотопные тождественному $\operatorname{rel} \partial S$.

Группа кос

$$B_n \cong \langle \sigma_1, \dots, \sigma_{n-1} \mid \sigma_i\sigma_j = \sigma_j\sigma_i, \ |i-j| \geqslant 2; \ \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1} \rangle$$

(образующие σ_i при таком ее задании называют *образующими Артина*) изоморфна группе $\mathrm{MCG}(B^2_{(n)})$. Классическим введением в эту область исследований может служить книга [2].

0.2. Проблема тождества. Для любой группы G представляют интерес вопросы: разрешима ли для G *проблема тождества* (ее можно сформулировать следующим образом: существует ли для G *алгоритм распознавания*, определяющий, представляет ли слово в данных образующих и обратных к ним единичный элемент

Работа частично поддержана грантом поддержки ведущих научных школ, No. 00-15-96024.

группы) и если да, то насколько сложны соответствующие алгоритмы. Для групп классов отображений поверхностей и, в частности, для группы кос проблема тождества разрешима и известно несколько существенно различных алгоритмов распознавания.

Особый интерес представляют *унифицированные алгоритмы* распознавания и их сложность. Унифицированный алгоритм решает проблему тождества для каждой группы G_n из некоторой последовательности групп $\{G_i\}$, n при этом играет роль входного параметра для алгоритма.

0.3. Проблема сравнения. Если на группе G задан некоторый порядок, то естественно возникает *проблема сравнения* в этом порядке, которую в случае линейного порядка можно рассматривать как усложненную проблему распознавания. Интересны не произвольные, а инвариантные (хотя бы односторонне) по отношению к групповой операции порядки. Деорнуа [4] описал (алгебраическими методами) односторонне-инвариантный порядок $\prec_{\mathrm{D}}$ на группе B_n (двусторонне-инвариантного порядка для кос не существует) и *алгоритм сравнения* для $\prec_{\mathrm{D}}$. В работе [12] геометрическими методами описан обширный класс односторонне-инвариантных порядков на группе классов отображений поверхности с непустой границей (в случае проколотого диска включающий $\prec_{\mathrm{D}}$). В [11] указан метод построения алгоритмов сравнения для некоторых из этих порядков. MCG-группа поверхности с пустым краем обладает кручением, и, следовательно, линейных инвариантных порядков на ней не существует.

Скорости известных алгоритмов обсуждаются в §1.

0.4. Основные результаты работы.

1) Описано широкое семейство односторонне-инвариантных порядков на MCG-группе поверхности с непустым краем, включающее, в частности, все инвариантные порядки, систематически описанные в литературе ранее.

2) Описан способ построения алгоритмов сравнения (имеющих квадратичную сложность) для всех *конечных* порядков на MCG-группе любой поверхности с непустым краем. (Множество конечных порядков включает в себя, в частности, все порядки, алгоритмы сравнения для которых были описаны ранее.)

3) Дан унифицированный алгоритм сравнения $A_{\prec_{\mathrm{D}}}(B_n)$ для порядка Деорнуа на группе кос, RAM$_s$-, RAM$_\ell$- и TM-сложности

которого равны

$$O(|w|+n), \quad O(|w|^2+|w|n+n\log n) \quad \text{и} \quad O(|w|^2n+|w|n\log n),$$

соответственно. Компьютерная реализация алгоритма $A_{\prec_{\mathrm{D}}}(B_n)$ имеет сложность $O(|w|^2+|w|n)$. Мультипликативные постоянные (см. приложение 1) всех указанных сложностей малы. Заметим, что текст реализующей алгоритм программы целиком помещается на одной странице. Мы полагаем, что это самый простой и самый быстрый алгоритм (как сравнения, так и распознавания) для кос, известный на данный момент.

4) Указан также алгоритм сравнения $A'_{\prec_{\mathrm{D}}}(B_n)$ с практической точки зрения менее удобный, чем $A_{\prec_{\mathrm{D}}}(B_n)$, но имеющий меньшую TM-сложность. Его RAM$_s$-, RAM$_\ell$- и TM-сложности равны

$$O(|w|+n), \quad O(|w|^2+|w|n+n\log n) \quad \text{и} \quad O(|w|^2n),$$

соответственно (ср. с таблицей §1).

5) Приведены примеры топологического описания нормальных форм для групп кос и указаны алгоритмы их нахождения.

0.5. Структура статьи. В §1 обсуждается время работы известных алгоритмов сравнения и распознавания для MCG-групп.

В §2 определяются используемые топологические понятия и описывается семейство порядков на MCG-группе поверхности с непустым краем.

В §3 описывается способ получения алгоритмов сравнения для конечных порядков на MCG-группе поверхности с непустым краем.

В §4 приводится унифицированный алгоритм сравнения для порядка Деорнуа на группе кос.

В §5 проводится оценка сложности унифицированных алгоритмов сравнения для порядка Деорнуа на группе кос.

В приложении 1 определяются используемые в работе понятия теории сложности вычислений.

В приложении 2 обсуждается вопрос о нормальных формах и приводятся примеры нормальных форм.

0.6. Благодарности. Я благодарю А. М. Вершика, привлекшего мое внимание к проблеме нормальных форм, за многочисленные ценные обсуждения. Я благодарен Э. А. Гиршу за обсуждение вопросов, касающихся различий между разными определениями

вычислительной сложности алгоритмов. Я благодарю своего научного руководителя Н. Ю. Нецветаева за интерес и внимание к моей работе и за помощь при написании статьи.

§1. Сложность известных алгоритмов распознавания и сравнения

1.1. Случай одной группы. Определение понятий, касающихся сложности алгоритмов, см. в приложении 1. В этих определениях мы следуем [1].

Сложность алгоритмов, упоминаемых в этом пункте, одна и та же во всех рассматриваемых нами моделях (RAM_s, RAM_ℓ и TM), поэтому будем говорить просто о *сложности*.

Впервые вопрос о времени работы алгоритма распознавания для группы кос был поставлен Терстоном [5]. Он указал, что сложность известных к тому времени алгоритмов экспоненциальна (либо не оценена, но при первом рассмотрении кажется таковой), и привел алгоритмы распознавания для групп B_n и $\mathrm{MCG}(S^2_{(n)})$ (здесь S^2 – сфера), имеющие квадратичную ($O(|w|^2)$) сложность.

Мошер [9], используя развитые в [5] идеи автоматных групп, указал метод построения алгоритма распознавания, имеющего квадратичную сложность, для MCG-группы любой поверхности.

В [11] для порядка $\prec_{\mathrm{D}}$ и для некоторых односторонне-инвариантных порядков на MCG-группе любой поверхности с непустым краем на основе результатов Мошера получены алгоритмы сравнения, имеющие квадратичную сложность.

Алгоритмы распознавания для групп кос и для MCG-групп поверхностей, имеющие квадратичную сложность, получены также и другими методами [3, 7].

1.2. Сложность унифицированных алгоритмов распознавания для групп кос. Насколько известно автору, наилучшие результаты достигнуты в работах [5, 3, 7]. Вот как выглядит сводная таблица порядков асимптотической временно́й сложности (определение см. в приложении 1) представленных там алгоритмов:

	RAM_s	RAM_ℓ	TM
Т [5]	$\lvert w\rvert^2 n\log n$	$\lvert w\rvert^2 n\log n\log(\lvert w\rvert n)$	$\lvert w\rvert^2 n\log^2 n$
Б–К–Л [3]	$\lvert w\rvert^2 n$	$\lvert w\rvert^2 n\log(\lvert w\rvert n)$	$\lvert w\rvert^2 n\log n$
Х-Т [7]	$\lvert w\rvert n$	$\lvert w\rvert^2 n+\lvert w\rvert n\log n$	$\lvert w\rvert^2 n+\lvert w\rvert n\log n$

Поскольку при компьютерной реализации каждое из значений n и $|w|$ разумно считать помещающимся в одной ячейке памяти и при оценке сложности величины порядка $\log n$, $\log|w|$ можно не учитывать, то справедливо следующее утверждение:

Компьютерная реализация любого из трех указанных унифицированных алгоритмов имеет асимптотическую временну́ю сложность $O(|w|^2n)$.

§2. Кривые и порядки

2.1. Изотопические классы кривых на поверхности. Зафиксируем пару (S,P_{in}). Если $\partial S\neq\varnothing$, зафиксируем $P_\partial\subset\partial S$ – конечное множество отмеченных точек, такое, что на каждом компоненте границы отмечена хотя бы одна точка. Положим $P=P_{\text{in}}\cup P_\partial$. Точки множества P_∂ и компоненты связности $\partial S\smallsetminus P_\partial$ будем называть *элементами границы* S.

Говоря о *замкнутой кривой* на (S,P), будем иметь в виду простую, нетривиальную (т.е. не ограничивающую диск в $S\smallsetminus P$) замкнутую кривую на $S\smallsetminus P$.

Незамкнутой кривой или *дугой* на (S,P) будем называть отображение
$$f\colon(I,\partial I,\operatorname{int}I)\to(S,P\cup\partial S,S\smallsetminus P),$$
являющееся вложением на $\operatorname{int}I$ и *нетривиальное*. (Последнее означает, что в удвоении $\operatorname{Dopp}(S,\partial S,P,f(I))$ кривая $\operatorname{Dopp}(f(I))$ не может ограничивать дисков, не содержащих внутри точек множества $\operatorname{Dopp}(P)$.)

К одному изотопическому классу отнесем кривые, которые переходят друг в друга при гомеотопиях $S\operatorname{rel}P$ (концы незамкнутой кривой, находящиеся на компоненте связности $\partial S\smallsetminus P_\partial$, могут по этой компоненте “двигаться”). Ниже изотопические классы кривых называются *классами кривых*. При рассмотрении кривых, снабженных ориентацией, возникают изотопические классы ориентированных кривых.

Будем говорить, что две кривые (возможно, ориентированные) находятся в *натянутом положении*, если изотопиями их нельзя перевести в положение, при котором они будут иметь меньшее количество точек пересечения (pulling tight).

Индекс пересечения $I(h_1, h_2)$ классов кривых h_1 и h_2 определим как количество точек пересечения в $S \setminus P$ их представителей, находящихся в натянутом положении. *Непересекающимися* назовем классы, индекс пересечения которых равен нулю. Индекс пересечения двух наборов классов кривых $I(\{h_i\}, \{h'_j\})$ определим как $\sum_{i,j} I(h_i, h'_j)$.

2.2. Порядки на MCG-группе поверхности с непустым краем. Пусть $\partial S \neq \varnothing$. Фиксируем ориентацию на всех компонентах ∂S. Множество классов направленных дуг на (S, P) с началом на элементе границы x обозначим $\vec{\Gamma}_x$.

Порядок на классах дуг. На множестве $\vec{\Gamma}_x$ существует пара канонических взаимно-обратных линейных порядков $(<_{x\pm})$ – они связаны с поднятием дуг в универсальное накрывающее $S_{(\cdot)}$. Коротко описать эти порядки можно следующим образом: рассмотрим пару ориентированных дуг, представляющих два различных элемента $\vec{\Gamma}_x$ и находящихся в натянутом положении; тот класс, дуга которого при выходе из x находится левее (это понятие дает введенная ориентация границы), будем считать бóльшим в порядке $<_{x+}$ и меньшим – в порядке $<_{x-}$.

Конечные порядки на MCG-группе. Пусть $\vec{H}^{\pm} = \{\vec{h}_i^{\delta_i}\}$ $(\vec{h}_i \in \vec{\Gamma}_{x_i}, \delta_i = \pm)$, – конечный упорядоченный набор снабженных знаками классов ориентированных дуг. Будем называть такие наборы *упорядочивающими системами*.

Определим порядок $\prec_{\vec{H}^{\pm}}$ на группе $\mathrm{MCG}(S_{(\cdot)})$ следующим образом:

Для $\alpha, \beta \in \mathrm{MCG}(S_{(\cdot)})$ положим

$$\alpha \prec_{\vec{H}^{\pm}} \beta, \text{ если } (\vec{h}_j)\alpha = (\vec{h}_j)\beta \text{ при } j < i \text{ и } (\vec{h}_i)\alpha <_{x_i\delta_i} (\vec{h}_i)\beta.$$

Так как элементы группы $\mathrm{MCG}(S_{(\cdot)})$ действуют в множестве $\vec{\Gamma}_x$, сохраняя порядки $<_{x\pm}$, порядок $\prec_{\vec{H}^{\pm}}$ (частичный либо линейный в зависимости от $\vec{H}^{\pm}$) будет право-инвариантным. Порядки, задаваемые упорядочивающими системами, будем называть *конечными*. (Мы заимствуем этот термин из [12], хотя наше множество конечных порядков шире.) От выбора множества P_∂ и ориентации границы множество конечных порядков $S_{(\cdot)}$ не зависит.

2.3. Бесконечные порядки на MCG-группе поверхности с непустым краем. Пусть $\tilde{\Gamma}_x$ – пополнение $\vec{\Gamma}_x$ в любом из двух канонических порядков. На $\tilde{\Gamma}_x$ продолжаются порядки $<_{x\pm}$ и действие элементов группы $\mathrm{MCG}(S_{(\cdot)})$, поэтому имеет смысл рассматривать упорядочивающие системы, включающие элементы $\tilde{\Gamma}_x$. Порядки, задаваемые такими системами и не являющиеся конечными, назовем *бесконечными* (ср. [12]).

§3. Алгоритмы сравнения для конечных порядков

Пусть дана пара (S, P), где S – поверхность с непустым краем.

3.1. Диаграммы и кодировки. Для построения алгоритмов определим следующие объекты:

Диаграмма – неупорядоченный набор непересекающихся классов неориентированных дуг, включающий лишь классы дуг, оба конца которых лежат на $\partial S \setminus P_\partial$. (Диаграмму можно представлять, как одномерное подмногообразие в $S \setminus P$.)

Диаграмму, в орбите которой MCG-группа действует свободно, будем называть *распознающей*. Любая диаграмма может быть дополнена до распознающей.

Предложение. *Любой конечный порядок может быть задан упорядочивающей диаграммой – упорядочивающей системой, дуги которой (при отбрасывании порядка и ориентации) образуют диаграмму. Существует эффективный способ нахождения такой диаграммы для заданной упорядочивающей системы. Конечный порядок является линейным тогда и только тогда, когда задающие его упорядочивающие диаграммы являются распознающими.*

Это предложение можно доказать, например, при помощи методов работы [12]. Результаты той же работы дают возможность заключить, что рассмотрение дуг с самопересечениями не дает более широкого класса порядков.

Для представлений диаграмм будем использовать их кодировки в триангуляциях:

Триангуляция – максимальный набор различных непересекающихся классов неориентированных дуг (*ребер* триангуляции), включающий лишь классы дуг, оба конца которых лежат в P. (Ср. [9], – идеальная триангуляция.) Любая триангуляция содержит все *граничные* (целиком лежащие на ∂S) дуги. В случае диска с пустым множеством отмеченных точек триангуляция состоит из пустого множества ребер, но и MCG-группа в этом случае тривиальна.

Кодировкой $T(D)$ диаграммы D в триангуляции T будем называть набор индексов пересечения $\{I(h,D)\}_{h\in T}$.

Доказательство следующего предложения стандартно (см., например, [9]).

Предложение. *Кодировка $T(D)$ любой диаграммы D в любой триангуляции T однозначно определяет D.*

3.2. Элементарная вычислимость. Следующие функции назовем *элементарными*:

$0, 1, \operatorname{sign}(z), \pm z, |z| \colon \mathbb{Z} \to \mathbb{Z}$;

$z_1 + z_2, \max(z_1, z_2), z_1 \operatorname{sign}(z_2) \colon \mathbb{Z}\times\mathbb{Z} \to \mathbb{Z}$;

$\mathrm{pr}_i \colon \mathbb{Z}^r \to \mathbb{Z}, \quad (z_1, \dots, z_r) \mapsto z_i$.

(Список можно сократить.)

Если $T = (h_1, \dots, h_{|T|})$ – триангуляция для (S, P), то верны два следующих утверждения:

Принцип пересчета кодировок. *Для любого $\alpha \in \mathrm{MCG}(S_{(\cdot)})$ существует процедура пересчета – отображение*

$$\Psi_{(S,T,\alpha)} \colon \mathbb{Z}^T \to \mathbb{Z}^T,$$

такое, что каждая его компонента (т.е. функция $\mathrm{pr}_i \circ \Psi_{(S,T,\alpha)} \colon \mathbb{Z}^T \to \mathbb{Z}^{h_i}$) – композиция элементарных функций и $T(D\alpha) = \Psi_{(S,T,\alpha)} T(D)$.

Принцип вычислимости порядка. *Для любой упорядочивающей диаграммы $\vec{D}^{\pm}$ существует функция сравнения*

$$\Theta_{(S,T,\vec{D}^{\pm})} \colon \mathbb{Z}^T \to \{-1, 0, +1\},$$

являющаяся композицией элементарных функций, такая, что для любого $\alpha \in \mathrm{MCG}(S_{(\cdot)})$

$$\Theta_{(S,T,\vec{D}^{\pm})}(T(\vec{D}^{\pm}\alpha)) = \begin{cases} -1, & \text{если } \alpha \prec_{\vec{D}^{\pm}} e, \\ +1, & \text{если } e \prec_{\vec{D}^{\pm}} \alpha, \\ 0, & \text{если } \alpha \text{ и } e \text{ несравнимы} \\ & \text{(равны, когда } \prec_{\vec{D}^{\pm}} \text{ – линейный).} \end{cases}$$

3.3. Построение алгоритма. Пусть $\vec{D}^{\pm}$ – упорядочивающая диаграмма, T – триангуляция пары (S, P),

$$X^{\pm} = (x_1, \dots, x_m, x_1^{-1}, \dots, x_m^{-1})$$

– конечное множество образующих группы $\mathrm{MCG}(S_{(\cdot)})$ и обратных к ним. (Классы $X^{\pm}$ могут быть заданы, например, парами триангуляций (прообраз – образ), либо кривыми для скручиваний Дена и "движений" точек – так, чтобы у нас была возможность с помощью общеизвестных методов определять действие этих классов отображений на классах кривых.)

Для построения алгоритма найдем:

– кодировку $T(\vec{D}^{\pm})$,

– $2m$ процедур пересчета $\Psi_{(S,T,x_i^{\delta})}$,

– функцию сравнения $\Theta_{(S,T,\vec{D}^{\pm})}$.

Удобство выбранного представления заключается в том, что существует эффективный способ нахождения перечисленных объектов.

Алгоритм сравнения. При вводе слова $W = w_1 \dots w_k$ $(w_i \in X^{\pm})$ алгоритм сравнения, начиная с кодировки $T(\vec{D}^{\pm})$, переходит последовательно от кодировки $T(\vec{D}^{\pm}[w_1 \dots w_{i-1}])$ к $T(\vec{D}^{\pm}[w_1 \dots w_i])$, при помощи процедур $\Psi_{(S,T,w_i)}$. Получив после k таких операций кодировку $T(\vec{D}^{\pm}[W])$, алгоритм вычисляет значение

$$\Theta_{(S,T,\vec{D}^{\pm})}(T(\vec{D}^{\pm}[W])).$$

Полученное значение (± 1 или 0) – искомый ответ. ■

Замечание. Если нам необходимо решить для группы $\mathrm{MCG}(S_{(\cdot)})$ лишь проблему тождества, вместо упорядочивающей диаграммы $\vec{D}^{\pm}$ возьмем любую распознающую диаграмму D_0. Найдя кодировку $T(D_0[W])$ так же, как в описанном алгоритме сравнения, алгоритм распознавания должен сравнить ее с кодировкой $T(D_0)$. Если $T(D_0[W]) = T(D_0)$, то $[W] = e$; иначе $[W] \neq e$.

Сложность построенного алгоритма. *Полученный алгоритм сравнения (распознавания) имеет линейную* RAM_s*-сложность и квадратичные* RAM_ℓ*- и* TM*-сложности.*

(Очевидно.)

Нахождение быстрого алгоритма для заданных поверхности и порядка. Подобрав для заданной поверхности (и фиксированного конечного порядка) конфигурацию $(X^{\pm}, T, D_0(\vec{D}^{\pm}))$ таким образом, чтобы индексы взаимных пересечений классов кривых, представляющих все объекты конфигурации, были малы (образующие $X^{\pm}$ можно задавать кривыми, – см. выше), можно получить несложные процедуры пересчета (и функцию сравнения) и, соответственно, малые мультипликативные постоянные при асимптотической сложности получающегося алгоритма распознавания (сравнения).

3.4. Замечание. Похожим способом можно получить быстрые алгоритмы распознавания и для MCG-групп других поверхностей.

Так, почти все поверхности с пустым краем, но непустым множеством отмеченных точек, обладают непустыми триангуляциями, и их MCG-группа нетривиально действует в множестве кривых (исключением являются лишь нескольких простых поверхностей). Рассматривая вместо диаграмм наборы непересекающихся классов замкнутых кривых и так же кодируя их в триангуляциях, получим алгоритмы распознавания для MCG-групп таких поверхностей.

(Отметим, что наборы непересекающихся классов замкнутых кривых не могут быть распознающими, но всегда существуют небольшие *распознающие совокупности* таких наборов. Принцип пересчета кодировок и простота способов нахождения процедур пересчета сохраняются.)

§4. Алгоритм сравнения для порядка Деорнуа в группе кос

Обратимся к последовательности групп кос.

Как уже упоминалось, группа кос B_n изоморфна MCG-группе диска с n выколотыми точками. Фиксируем ориентацию на ∂B^2. На (B^2, P) ($|P_{\text{in}}| = n$, $|P_\partial| = 4$) рассматриваются следующие объекты:

1. Триангуляция T_n (см. рис. 1).

2. Система образующих $\Sigma_n = (\sigma_1, \dots, \sigma_{n-1})$, где σ_i – класс гомеоморфизма, меняющего местами концы ребра b_i (см. рис. 1), проводя их вдоль b_i так, чтобы точки “разминулись по часовой стрелке”. (Поставив в соответствие с каждым классом отобра-

жений σ_i образующую Артина σ_i группы кос B_n, получим стандартный изоморфизм $\mathrm{MCG}(B^2_{(n)}) \cong B_n$.)

3. Упорядочивающая диаграмма

$$\vec{D}_n = (\vec{d}_1^{\,+}, \vec{d}_2^{\,+}, \dots, \vec{d}_{n-1}^{\,+})$$

(см. рис. 2). $\vec{D}_n$ задает на группе $\mathrm{MCG}(B^2_{(n)}) \cong B_n$ порядок Деорнуа (см. [6]).

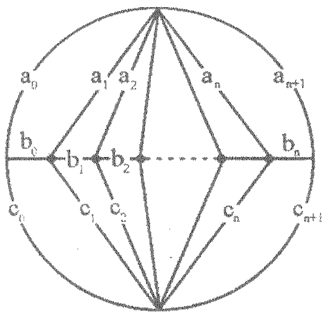

Рис. 1.

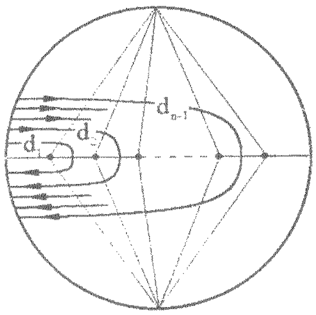

Рис. 2.

Унифицированный алгоритм сравнения $A_{\prec_{\mathrm{D}}}(B_n)$ для последовательности конфигураций $\{(\Sigma_i, T_i, \vec{D}_i)\}$ является унифицированным алгоритмом сравнения для порядка Деорнуа в группе кос.

Значения кодировки $T_n(\vec{D}_n)$ ясны из рис. 2. Выбранное соотношение образующих и триангуляции позволяет унифицировать процедуры пересчета $\Psi_{(B^2_{(n)}, T_n, \sigma_i^\delta)}$. (Рис. 3 иллюстрирует действие образующей на соответствующем участке триангуляции.) Функция сравнения $\Theta_{(B^2_{(n)}, T_n, \vec{D}_n)}$ описывается очень коротко (см. алгоритм).

Алгоритм $\mathbf{A}_{\prec_{\mathrm{D}}}(\mathbf{B_n})$:

(Входные параметры: индекс n и слово длины k в образующих Артина группы кос B_n и обратных к ним: $W = w_1 \dots w_k = \sigma_{i_1}^{\delta_1} \dots \sigma_{i_k}^{\delta_k}$.)

1) Зарезервировать массив для записи $3n+5$ неотрицательных целых чисел: $(ABC)_n = \{a_i\}_{1..(n+1)}$; $\{b_i\}_{1..n}$; $\{c_i\}_{1..(n+1)}$.
Записать в массив $(ABC)_n$ следующие значения:
При $1 \leqslant i \leqslant n$: $a_i := n - i$, $c_i := n - i$, $b_i := 1$;
значения остальных элементов – 0.
2) Последовательно для всех целых значений t от 1 до k:
Выполнить процедуру Ψ от параметров (i_t, δ_t).
3) Вычислить значение функции $\Theta(\{a_i\}_{1..(n+1)}; \{b_i\}_{1..n}; \{c_i\}_{1..(n+1)})$.
Если $\Theta = 0$, то $[W] = e$.
Если $\Theta = +1$, то $e \prec_{\mathrm{D}} [W]$.
Если $\Theta = -1$, то $[W] \prec_{\mathrm{D}} e$.
Конец алгоритма.

Процедура Ψ от параметров i $(1 \leqslant i \leqslant n-1)$ и δ $(\delta = \pm 1)$:
Если $\delta = +1$, то выполнить:
$b' := \max(b_{i-1} + c_{i+1},\ b_i + c_{i-1}) - c_i$;
$b'' := \max(a_{i+2} + b_i,\ a_i + b_{i+1}) - a_{i+1}$;
$a' := \max(a_{i-1} + b_i,\ a_i + b') - b_{i-1}$;
$c' := \max(c_{i+2} + b_i,\ c_{i+1} + b'') - b_{i+1}$;
$a_{i+1} := a_i$; $a_i := a'$;
$b_{i-1} := b'$; $b_{i+1} := b''$;
$c_i := c_{i+1}$; $c_{i+1} := c'$.
Если $\delta = -1$, то выполнить:
$b' := \max(a_{i-1} + b_i,\ a_{i+1} + b_{i-1}) - a_i$;
$b'' := \max(b_i + c_{i+2},\ b_{i+1} + c_i) - c_{i+1}$;
$a' := \max(b_i + a_{i+2},\ a_{i+1} + b'') - b_{i+1}$;
$c' := \max(b_i + c_{i-1},\ c_i + b') - b_{i-1}$;
$a_i := a_{i+1}$; $a_{i+1} := a'$;
$b_{i-1} := b'$; $b_{i+1} := b''$;

$c_{i+1} := c_i$; $c_i := c'$.

Конец процедуры Ψ.

Функция Θ:

Если все члены последовательности $\{a_i - c_i\}_{1..(n-1)}$ – нули, то $\Theta = 0$.

Иначе Θ равно знаку первого ненулевого члена $\{a_i - c_i\}_{1..(n-1)}$.

Конец функции Θ.

Пояснения к алгоритму. Массив переменных $(ABC)_n$ используется алгоритмом $A_{\prec_{\mathrm{D}}}(B_n)$ для записи значений кодировки, т.е. переменные массива принимают целые неотрицательные значения. В приведенной записи переменные массива поименованы так же, как соответствующие им ребра на рис. 1.

В процессе работы алгоритма в массив $(ABC)_n$ последовательно для $t = 0, 1, \ldots, k$ записываются кодировки $T_n(\vec{D}_n[W_t])$ образов $\vec{D}_n[W_t]$ диаграммы $\vec{D}_n$ $(W_t = w_1 w_2 \ldots w_t)$. Кодировка $T_n(\vec{D}_n[W_0]) = T_n(\vec{D}_n)$ записывается "вручную", переход к каждой следующей кодировке осуществляется унифицированной процедурой пересчета Ψ. Процедура Ψ использует целочисленные переменные a', b', b'', c', в которые записываются промежуточные значения.

Если M_t – максимум значений кодировки $T_n(\vec{D}_n[W_t])$ на ребрах T_n, то $M_{t+1} \leqslant 2M_t$. Поэтому при работе со словом длины k в образующих B_n в массиве будут храниться числа, длина которых (в битах) не превосходит $k + \log_2 n$.

Для простоты в приведенной записи алгоритма дано словесное описание функции сравнения.

§5. Оценка сложности алгоритма $A_{\prec_{\mathrm{D}}}(B_n)$ и алгоритм $A'_{\prec_{\mathrm{D}}}(B_n)$

Утверждение. *Алгоритм $A_{\prec_{\mathrm{D}}}(B_n)$ имеет следующие сложности:*

i) RAM$_s$-: $O(|w| + n)$;

ii) RAM$_\ell$-: $O((|w| + n)(|w| + \log n))$;

iii) TM-: $O(|w|n(|w| + \log n))$.

Доказательство. i) При обработке слова длины k в образующих B_n:

– запись в массив $(ABC)_n$ начальной кодировки $T_n(\vec{D}_n)$ производится RAM-машиной за $O(n)$ операций;

– при каждом обращении к процедуре Ψ машина совершает фиксированное количество операций, не зависящее от n и k;

– таким образом, при получении по кодировке $T_n(\vec{D}_n)$ кодировки $T_n(\vec{D}_n[W])$ производится $O(k)$ операций;

– при сравнении машина совершает $O(n)$ операций;

Суммируем:

$$O(n) + O(k) + O(n) = O(k+n).$$

ii) При обработке слова длины k в образующих B_n алгоритм использует $O(n)$ регистров RAM-машины и записывает в них числа, не превосходящие $2^k n$, поэтому время совершения каждой операции есть $O(k + \log n)$.
Умножаем:

$$O(k+n)O(k+\log n) = O((k+n)(k+\log n)).$$

iii) Алгоритм достаточно прост, и мы не будем подробно описывать работу реализующей его машины Тьюринга. Массив $(ABC)_n$ можно разместить, например, на одной из лент машины в порядке $a_0, c_0, b_0, a_1, c_1, \dots, a_{n+1}, c_{n+1}$ и совершать с ним все необходимые операции "естественным" образом. При обработке слова длины k в образующих B_n:

– запись начальной кодировки производится за $O(n \log n)$ шагов;

– длина записи текущей кодировки в любой момент – $O(n(k + \log n))$;

– переход от кодировки $T_n(\vec{D}_n[W_{t-1}])$ к кодировке $T_n(\vec{D}_n[W_t])$ занимает $O(n(k + \log n))$ шагов, – это время, в основном, тратится на сдвиг части массива и переход к записям значений, которые нужно изменить, само же изменение значений требует лишь $O(k + \log n)$ шагов;

– итого для получения кодировки $T_n(\vec{D}_n[W])$ по кодировке $T_n(\vec{D}_n)$ производится $O(kn(k + \log n))$ шагов;

– сравнение производится за $O(n(k + \log n))$ шагов;

Суммируем:

$$O(n \log n) + O(kn(k+\log n)) + O(n(k+\log n)) = O(kn(k+\log n)). \qquad \bullet$$

Алгоритм $A'_{\prec_{\mathrm{D}}}(B_n)$**.** Рассмотрим теперь конфигурацию, при которой индексы пересечений дуг упорядочивающей диаграммы и ребер триангуляции малы: Пример – конфигурация $(T'_n, \vec{D}'_n)$, изображенная на рис. 4. Обозначим построенный описанным выше способом унифицированный алгоритм для последовательности конфигураций $\{(\Sigma_i, T'_i, \vec{D}'_i)\}$ через $A'_{\prec_{\mathrm{D}}}(B_n)$. Асимптотические временны́е сложности в RAM-модели у $A'_{\prec_{\mathrm{D}}}(B_n)$ будут те же, что и у $A_{\prec_{\mathrm{D}}}(B_n)$, а ТМ-сложность – меньше.

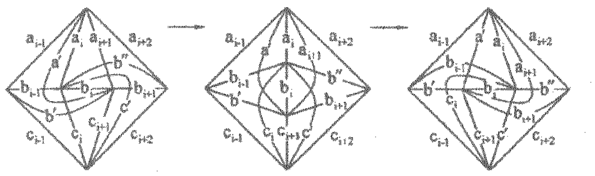

Рис. 3.

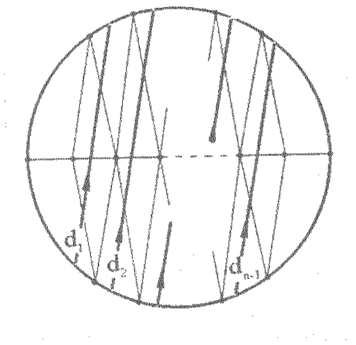

Рис. 4.

Утверждение. ТМ-*сложность алгоритма* $A'_{\prec_{\mathrm{D}}}(B_n)$ *равна* $O(|w|^2 n)$.

Доказательство. Пусть входные параметры представлены в унарной записи. При обработке слова длины k в образующих B_n:

– запись начальной кодировки производится за $O(n)$ шагов;

– длина записи текущей кодировки в любой момент – $O(nk)$;

– получение кодировки $T'_n(\vec{D}'_n[W_t])$ по $T'_n(\vec{D}'_n[W_{t-1}])$ – $O(nk)$ шагов;
– при нахождении кодировки $T'_n(\vec{D}'_n[W])$ совершается $O(knk)$ шагов;
– сравнение производится за $O(nk)$ шагов;

Итого:

$$O(n) + O(knk) + O(nk) = O(knk). \qquad \bullet$$

С практической точки зрения алгоритм $A_{\prec_{\mathrm{D}}}(B_n)$ удобнее, чем $A'_{\prec_{\mathrm{D}}}(B_n)$, – процедуры пересчета для $A'_{\prec_{\mathrm{D}}}(B_n)$ более громоздки, чем для $A_{\prec_{\mathrm{D}}}(B_n)$, и при компьютерной реализации алгоритма $A'_{\prec_{\mathrm{D}}}(B_n)$ мы получим при той же асимптотике, что и у $A_{\prec_{\mathrm{D}}}(B_n)$, бо́льшую мультипликативную постоянную.

Замечание. Похожим способом получаются быстрые унифицированные алгоритмы и для некоторых других последовательностей MCG-групп.

Приложение 1. Элементы теории сложности (ср. [1])

Среди множества критериев сложности алгоритма основным (по крайней мере, в нашем случае) считается временна́я сложность, описывающая продолжительность работы алгоритма как функцию от входных параметров.

Асимптотическая сложность. Говорят, что алгоритм A сравнения или распознавания имеет *асимптотическую временну́ю сложность* $O(f(|w|))$, если время его работы при подаче на вход слова w в образующих и обратных к ним не превышает $cf(|w|)$, где $|w|$ – длина слова w, c – некоторая константа. Асимптотическую временну́ю сложность мы иногда называем *асимптотической сложностью* или просто *сложностью*. Минимальную подходящую для оценки константу c будем называть *мультипликативной постоянной* при асимптотической сложности (обычно ее не вычисляют, а говорят о бо́льших и меньших мультипликативных постоянных).

В основном именно асимптотическая сложность определяет, насколько большие входные параметры за приемлемое время сможет обработать алгоритм (хотя, конечно, встречаются и случаи, когда у некоторого алгоритма мультипликативная постоянная при асимптотической сложности настолько велика, что для практического использования лучше подходит алгоритм с большей асимптотической сложностью, но меньшей мультипликативной постоянной).

Сложность унифицированного алгоритма определяют аналогично, используя функции от двух параметров – $|w|$ и n.

Вычислительные модели. Чтобы оценить сложность алгоритма, нужно рассмотреть его реализацию в некоторой *вычислительной модели*, для которой определено понятие "времени работы алгоритма". Сложности алгоритма в разных моделях могут существенно различаться. Одни модели при этом лучше отражают время работы алгоритма на реальном компьютере, чем другие.

В качестве вычислительных моделей мы рассматриваем многоленточные машины Тьюринга (TM-*модель*) и (random access memory) машины (RAM-*модель*). В описании этих моделей мы следуем [1].

TM-сложность. Временем работы машины Тьюринга считают количество шагов, сделанных машиной; сложность алгоритма в TM-модели будем называть TM-*сложностью*. TM-модель обычно хуже отражает реальную скорость работы алгоритма, чем RAM-модель.

RAM$_\ell$-сложность. RAM-машины, в отличие от TM, могут содержать в регистре памяти сколь угодно большое число и на каждом шаге обращаться к любому регистру. Два сколь угодно больших целых числа RAM машина складывает за один шаг. Для вычисления времени работы алгоритма складывают промежутки времени, требующиеся для выполнения каждой команды (шага). Наиболее точно отражает реальную ситуацию (т.е. скорость работы компьютерной программы) способ исчисления времени, использующий *логарифмический критерий* – время выполнения команды считают пропорциональным длине ее операндов (адреса регистров, их содержимое). Длина операнда i по понятным причинам в формулах проявляется как $\log i$. Сложность алгоритма в RAM-модели при таком способе исчисления времени будем называть RAM$_\ell$-*сложностью* (ℓ от log).

RAM$_s$-сложность. Сложность в RAM-модели, при расчете которой используется *равномерный временной критерий* (считается, что любая команда выполняется за единицу времени, т.е. время работы равно количеству совершенных операций), будем называть RAM_s-*сложностью* (s от step). RAM_s-сложность хорошо отражает время работы реального компьютера, когда алгоритм имеет дело с операндами, значения которых помещаются в одной ячейке компьютера, при больших же операндах она может быть значительно меньше реальной компьютерной.

RAM_s-сложность любого алгоритма не превышает его RAM_ℓ-сложности и TM-сложности. RAM_ℓ-сложность обычно (но не всегда) меньше TM-сложности.

Приложение 2. Нормальные формы для группы кос в образующих Артина

Известные методы решения проблем тождества и сравнения для MCG-групп и групп кос можно, в основном, разделить на алгебраические и топологические (геометрические). Первые основаны на поиске той или иной нормальной формы, представляющей заданный элемент рассматриваемой группы, вторые (к ним относится и метод, описываемый в этой работе) – на определении действия элемента группы на топологических объектах (нормальные формы при этом находить не требуется).

Нормальные формы представляют самостоятельный интерес. Описанный в этой работе подход дает возможность конструировать нормальные формы (и алгоритмы для их нахождения) для различных MCG-групп и систем образующих. Для фиксированной MCG-группы при этом можно получать нормальные формы, существенно различающихся по своим свойствам.

Покажем это, приведя в качестве примера две нормальные формы для образующих Артина группы кос.

Пример 1. Обратимся к конструкции $(\Sigma_n, T_n, \vec{D}_n)$, описанной в начале §4. Порядок нас сейчас не интересует, поэтому отбросим его, а также ориентацию элементов упорядочивающей диаграммы $\vec{D}_n$, и будем обозначать эту диаграмму через D_n. Положим

$$\Sigma_n^{\pm} = (\sigma_1, \dots, \sigma_{n-1}; \sigma_1^{-1}, \dots, \sigma_{n-1}^{-1}),$$

где $\sigma_i^{\pm}$ – образующие и обратные к ним.

Пусть D – диаграмма. Индекс пересечения $I(T_n, D)$ (он равен сумме значений кодировки $T_n(D)$ на всех ребрах T_n) будем называть *нормой* диаграммы D.

Будем говорить, что $\alpha \in \mathrm{MCG}(B^2_{(n)})$ *распутывает* диаграмму D, если $I(T_n, D\alpha) < I(T_n, D)$.

Распутывающая последовательность для диаграммы $D_n\alpha$ – слово $w_1 w_2 \dots w_k$, где $w_i \in \Sigma_n^{\pm}$, такое, что w_t распутывает $D_n\alpha w_1 \dots w_{t-1}$, а $D_n\alpha w_1 \dots w_k = D_n$, т.е. $\alpha = w_k^{-1} \dots w_1^{-1}$.

Предложение. *Пусть $\alpha \in \mathrm{MCG}(B^2_{(n)})$, $\alpha \neq e$. Тогда хотя бы один из элементов $\Sigma_n^{\pm}$ распутывает диаграмму $D_n\alpha$. Саму диаграмму D_n не распутывает ни один элемент группы $\mathrm{MCG}(B^2_{(n)})$.*

Это ясно из простых геометрических соображений.

Следствие. *При любом $\alpha \in \mathrm{MCG}(B^2_{(n)})$ множество распутывающих последовательностей для $D_n\alpha$ непусто.*

Чтобы получить нормальную форму, достаточно указать правило, сопоставляющее каждому элементу определенную распутывающую последовательность.

Например, пусть $\prec$ – порядок на $\Sigma_n^{\pm}$, а $RP_{\prec}(\alpha)$ – распутывающая диаграмму $D_n\alpha$ последовательность, минимальная (в индуцированном порядком $\prec$ лексикографическом порядке) среди всех последовательностей, распутывающих $D_n\alpha$.

Тогда слово, обратное к $RP_{\prec}(\alpha)$ (обозначим его $NF_{\prec}(\alpha)$), – нормальная форма для α.

Простой и быстрый алгоритм вычисления $NF_{\prec}(\alpha)$ можно получить, например, используя унифицированную процедуру пересчета Ψ (см. §4).

Пример 2. Возьмем тот же стандартный изоморфизм $B_n \cong \mathrm{MCG}(B^2_{(n)})$, систему $\Sigma_n^{\pm}$ и триангуляцию T_n, но переопределим понятия диаграммы, кодировки (соответственно) и распутывания.

Диаграммы. Рассмотрим набор классов кривых

$$H_n = (h_1 = c_1, \dots, h_n = c_n)$$

($c_1, \dots, c_n$ – ребра T_n, см. рис. 1). *Диаграммами* будем называть элементы B_n-орбиты набора H_n. Определение кодировки для "новых" диаграмм дословно сохраняется. Предложение об однозначном определении кодировкой диаграммы и принцип пересчета кодировок выполняются (хотя процедуры пересчета станут более громоздкими).

Распутывание. Предварительные замечания: Для произвольного класса дуг h и триангуляции T' дуга – представитель класса h, находящаяся в натянутом положении по отношению ко всем ребрам триангуляции T', пересекает их в определенном порядке, однозначно определяемом h. Вместо слов "любой представитель класса кривых диаграммы, находящийся в натянутом положении относительно всех ребер триангуляции" будем писать просто "дуга диаграммы". Точки множества $P_{\mathrm{in}} \subset B^2$ обозначим $p_1, \ldots, p_n$ – слева направо на рис. 1.

Распутывание на этот раз определим лишь для элементов $\Sigma_n^{\pm}$. Будем говорить, что:

– σ_i *распутывает* диаграмму H, если дуга диаграммы H, начинающаяся в точке p_i, выходя из p_i, первым из ребер триангуляции T_n пересекает ребро a_{i+1}, а дуга, начинающаяся в точке p_{i+1}, выходя из p_{i+1}, первым пересекает ребро c_i, либо никаких ребер не пересекает вообще (т.е. равна ребру c_{i+1}).

– σ_i^{-1} *распутывает* H, если дуга диаграммы H, начинающаяся в p_i, выходя из p_i, первым пересекает ребро c_{i+1}, либо ребер не пересекает (равна c_i), а дуга, начинающаяся в p_{i+1}, выходя из p_{i+1}, первым пересекает ребро a_i.

Стабильная нормальная форма $NF'_{\prec}(\alpha)$. *Распутывающие последовательности* определим соответственно с новым определением распутывания. Предложение о распутывающем элементе и его следствие о существовании распутывающей последовательности остаются верными.

Воспользуемся порядком $\prec$. Минимальную (в индуцированном $\prec$ лексикографическом порядке) распутывающую (в новом определении) последовательность для $D_n\alpha$ обозначим $RP'_{\prec}\alpha$.

"Перевернем" способ определения нормальной формы, использованный в первом примере: пусть $NF'_{\prec}(\alpha) = RP'_{\prec}(-\alpha)$. $NF'_{\prec}(\alpha)$, – нормальная форма для α.

Вышеописанные методы доставляют простой алгоритм нахождения $NF'_{\prec}(\alpha)$.

Замечание. Как указано в [15], при изучении границы группы и асимптотического поведения траекторий случайных блужданий по группе представляют интерес нормальные формы, обладающие определенными свойствами типа стабильности.

Нормальная форма $NF'_{\prec}(\alpha)$ является примером такой "удобной" формы для группы кос.

Литература

1. А. Ахо, Дж. Хопкрофт, Дж. Ульман, *Построение и анализ вычислительных алгоритмов.* М., Мир, 1979.
2. J. S. Birman, *Braids, Links, and Mapping Class Groups.* Ann. of Math. Stud., Vol 82, Princeton, N.J., Princeton Univ. Press, 1974.
3. J. S. Birman, K. H. Ko, S. J. Lee, *A new approach to the word and conjugacy problem in the braid groups.* — Adv. Math. **139** (1998), 322–353.
4. P. Dehornoy, *A fast method of comparing braids.* — Adv. Math. **125** (1997), 200–235.
5. D. Epstein et al., *Word Processing in Groups.* Jones and Bartlett Publs., 1992.
6. R. Fenn, M. T. Greene, D. Rolfsen, C. Rourke, B. Wiest, *Ordering the braid groups.* — Pacific J. Math. **191** (1999), 41–74.
7. H. Hamidi-Tehrani, *On complexity of the word problem in braid groups and mapping class groups.* preprint.
8. L. Mosher, *Classification of pseudo-Anosovs.* Low Dimensional Topology and Kleinian Groups. 1986, pp. 13–75.
9. L. Mosher, *Mapping class groups are automatic.* — Ann. Math. **142** (1995), 303–384.
10. L. Mosher, *A user's guide to the mapping class group: Once punctured surfaces.* Geometric and computational perspectives on infinite groups. DIMACS Ser. Discrete Math. Theoret. Comput. Sci., Vol. 25, 1996, pp. 101–174.
11. C. Rourke, B. Wiest, *Order automatic mapping class groups.* — Pacific J. Math. **194** (2000), 209–227.
12. H. Short, B. Wiest, *Orderings of mapping class groups after Thurston.* preprint.
13. S. Kaplan, M. Teicher, *Solving the braid word problem via the fundamental group.* preprint.
14. D. Garber, S. Kaplan, M. Teicher, *A new algorithm for solving the word problem in braid groups.* preprint.
15. А. М. Вершик, *Динамическая теория роста в группах: энтропия, граница, примеры.* — Успехи мат. наук **55**, No. 4 (2000), 59–129.

Malyutin A. V. Fast algorithms for identification and comparison of braids.

A method for constructing various algorithms for mapping-class groups (e.g., for the braid group) is presented, and a family of one-side invariant orders on the mapping-class group of a surface with boundary is described. A method for constructing comparison algorithms for all finite orders on the mapping-class group of any surface with boundary is described, a fast and simple comparison algorithm for the Dehornois order on the braid group is presented, examples of normal forms for braid groups are given, and algorithms for finding the forms are indicated.

С.-Петербургский
государственный
университет

Поступило 31 декабря 2000 г.