ISSN 0132-3474

Номер 5 Сентябрь - Октябрь 2023

ПРОГРАММИРОВАНИЕ

www.sciencejournals.ru

СОДЕРЖАНИЕ

Номер 5, 2023

ПАРАЛЛЕЛЬНОЕ И РАСПРЕДЕЛЕННОЕ ПРОГРАММИРОВАНИЕ

АНАЛИЗ ДАННЫХ

КОМПЬЮТЕРНАЯ АЛГЕБРА

CONTENTS

No. 5, 2023

PARALLEL AND DISTRIBUTED SOFTWARE

DATA ANALYSIS

COMPUTER ALGEBRA

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 3–18

ПАРАЛЛЕЛЬНОЕ И РАСПРЕДЕЛЕННОЕ ПРОГРАММИРОВАНИЕ

УДК 519.6

ДВАДЦАТЬ ФУНКЦИЙ ПОДОБИЯ ДВУХ КОНЕЧНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

© 2023 г. И. Бурдонов[a],*, А. Максимов[a],**

[a] *Институт системного программирования РАН им. В.П. Иванникова*
109004, г. Москва, ул. А. Солженицына, д. 25, Россия

**E-mal: igor@ispras.ru*

***E-mal: andrew@ispras.ru*

Поступила в редакцию 09.01.2023 г.
После доработки 16.02.2023 г.
Принята к публикации 21.03.2023 г.

В статье рассматриваются различные числовые функции, определяющие степень “похожести” двух заданных конечных последовательностей. Эти меры подобия основаны на определяемом нами понятии вложения в последовательность. Частным случаем такого вложения является обычная подпоследовательность (subsequence). Другие случаи дополнительно требуют равенства расстояний между соседними символами подпоследовательности в обеих последовательностях. Это является обобщением понятия отрезка последовательности (substring), в котором эти расстояния единичны. Дополнительно может требоваться равенство расстояний от начала последовательностей до первого символа вложения или от последнего символа вложения до конца последовательностей. Кроме этих двух последних случаев, вложение может входить в последовательность несколько раз. В литературе используются такие функции как число общих вложений или числа пар вхождений вложений в последовательности. Кроме них, мы вводим еще три функции: сумма длин общих вложений, сумма минимумов числа вхождений общего вложения в обе последовательности и функция подобия на основе наибольшего по числу символов общего вложения. Всего рассматриваются 20 числовых функций, для 17 из которых предложены алгоритмы (в том числе новые) полиномиальной сложности, еще для двух функций алгоритмы имеют экспоненциальную сложность с уменьшенным показателем степени. В Заключении дается краткая сравнительная характеристика этих вложений и функций.

Ключевые слова: анализ последовательностей, общие подпоследовательности, наибольшие и максимальные общие подпоследовательности, каноническое вложение, соответствующие совместные вложения, комбинаторные алгоритмы для подпоследовательностей и вложений, аксиомы подобия

DOI: 10.31857/S0132347423050035, **EDN:** ZYAIZN

1. ВВЕДЕНИЕ

Анализ последовательностей широко используется в социальных, управленческих, политических, демографических, психологических науках, в химии, биоинформатике и при обработке текстов. Последние десятилетия появилось много работ, посвященных этой тематике [1–6]. Применяются различные метрики и меры сходства (подобия) последовательностей [4, 5].

В этой статье мы рассматриваем различные числовые функции, определяющие степень “похожести” двух заданных конечных последовательностей. Эти меры подобия основаны на определяемом нами понятии вложения в последовательность. Частным случаем такого вложения является подпоследовательность. Другие случаи учитывают дополнительно расстояния между символами подпоследовательности в обеих последовательностях. Например, последовательности “МЕТРИКА” и “МАРОККО” имеют наибольшую общую подпоследовательность “МРК”, с учетом расстояний между символами “Р-К”, с учетом расстояний между символами и от последнего символа вложения до конца последовательности “К-”, с учетом расстояний между символами и от начала последовательности до первого символа вложения “М----К”.

Понятие вложения вводится с использованием пустого символа, не принадлежащего алфавиту заданных последовательностей. Всего вводятся пять типов вложения: *E*-вложение получается заменой некоторых символов пустым символом, *L*-вложение получается из *E*-вложения удалением пустого префикса, *R*-вложение получается из *E*-вложения удалением пустого постфикса, *O*-вложение получается из *E*-вложения удалением пу-

стых префикса и постфикса, наконец, *A*-вложение получается из *E*-вложения удалением всех пустых символов, что совпадает с понятием подпоследовательности. Мнемоника обозначения вложений образована от английских слов: *E* – **E**mpty symbol (пустой символ), далее указание места, где пустых символов нет (there are no empty symbols): *L* – on the **L**eft (слева), *R* – on the **R**ight (справа), *O* – **O**utside (снаружи, т.е. слева и справа), *A* – **A**nywhere (в любом месте, т.е. нигде нет пустых символов).

Каждое вложение может иметь несколько вхождений в последовательность, т.е. несколько *E*-вложений, из которых данное вложение получается удалением префикса, постфикса, префикса и постфикса или всех пустых символов. Например, подпоследовательность “МРК” входит один раз в последовательность “МЕТРИКА” (соответствующее *E*-вложение “М--Р-К-”) и два раза в последовательность “МАРОККО” (соответствующие *E*-вложения “М-Р-К--” и “М-Р--К-”. Для вложений, которые могут содержать пустые символы, определяется понятие μ-длины как число непустых символов (для *A*-вложения совпадает с длиной подпоследовательности).

Для каждого из четырех типов вложения (*L*, *R*, *O* и *A*) определяются пять функций: 0) число общих вложений, 1) сумма μ-длин общих вложений, 2) сумма минимумов чисел вхождения общих вложений в заданные последовательности, 3) сумма произведений чисел вхождения общих вложений в заданные последовательности, а также 4) мера похожести, основанная на наибольшей (по длине) общей подпоследовательности (*longest common subsequence*, *lcs*).

Некоторые из этих двадцати функций хорошо известны, например, число общих подпоследовательностей (число общих *A*-вложений) или сумма произведений чисел вхождения общих подпоследовательностей в заданные последовательности (сумма произведений чисел *E*-вложений общих *A*-вложений в заданные последовательности) [3]. Другие функции, особенно учитывающие “расстояния” между символами, мы вводим в данной статье.

После настоящего Введения в разделе 2 вводятся основные понятия и обозначения. В разделе 3 рассматривается оптимизация общая для всех типов вложений: замена пустым символом тех символов, которые входят только в одну из двух последовательностей. Далее в разделах 4–7 рассматриваются четыре типа вложений *L*, *R*, *O* и *A*, и для каждого типа вложения – алгоритмы вычисления пяти функций 0, 1, 2, 3, 4. В Заключении подводятся итоги и намечаются направления дальнейших исследований.

2. ОПРЕДЕЛЕНИЯ И ОБОЗНАЧЕНИЯ

Для целых чисел i и j будем обозначать: $i..j = \{i, i+1, ..., j\}$, если $i \le j$, $i..j = \varnothing$, если $i > j$.

Конечная последовательность в алфавите H длиной $m \ge 0$ – это инъекция множества $1..m$ в множество H: $1..m \to H$. Множество конечных последовательностей в алфавите H обозначим H^*. Пустую последовательность (длины 0, пустая инъекция) обозначим (). Для непустой конечной последовательности x i-й элемент последовательности, $i \in 1..|x|$, обозначим $x_i = x(i)$. Отрезок $x_i, x_{i+1}, ..., x_j$ для $1 \le i \le j \le |x|$ обозначим $x[i..j]$. Для $i > j$ определим $x[i..j] = ()$. Префикс обозначим как $x[j] = x[1..j]$, пустой префикс (длины 0) определен, в том числе, и для пустой последовательности $()[0] = ()$. Конечная последовательность из $k \ge 0$ повторений символа $h \in H$ будем обозначать h^k: $|h^k| = k \;\&\; \forall\, i = 1..|k| h^k_i = h$. Также вместо h^1 будем писать просто h.

Конкатенация xy конечных последовательностей x и y определяется условиями: $|xy| = |x| + |y|$, $\forall\, i \in 1..|x|\ (xy)_i = x_i$ и $\forall\, j \in 1..|y|\ (xy)_{|x|+j} = y_j$. Нам также понадобится конкатенация пары (X, Y) конечных множеств конечных последовательностей с парой (z, t) конечных последовательностей, которую определим так: $(X, Y)(z, t) = \{(xz, yt) : (x, y) \in (X, Y)\}$. Будем считать, что в выражениях операция конкатенации приоритетнее операций над множествами (объединение, пересечение и разность).

Композицию функций f и g будем обозначать fg.

Введем *пустой символ* $\varepsilon \notin H$ и обозначим $H_\varepsilon = H \cup \{\varepsilon\}$.

Обозначим множество конечных последовательностей в алфавите H_ε:

• не начинающихся пустым символом $L(H) = \{v \in H_\varepsilon^* : |v| > 0 \Rightarrow v_1 \ne \varepsilon\}$;

• не заканчивающихся пустым символом $R(H) = \{v \in H_\varepsilon^* : |v| > 0 \Rightarrow v_{|v|} \ne \varepsilon\}$;

• не начинающихся и не заканчивающихся пустым символом

$$O(H) = \{v \in H_\varepsilon^* : |v| > 0 \Rightarrow v_1 \ne \varepsilon \;\&\; v_{|v|} \ne \varepsilon\} = L(H) \cap R(H).$$

Введем операции удаления пустых символов из конечной последовательности в алфавите H_ε:

• *удаление префикса пустых символов* λ: $H_\varepsilon^* \to L(H)$ определяется условием:

$$\forall v \in H_\varepsilon^* \lambda(\varepsilon \cdot v) = \lambda(v) \;\&\; \forall u \in L(H) \lambda(u) = u;$$

• *удаление постфикса пустых символов* ρ: $H_\varepsilon^* \to R(H)$ определяется условием:

$$\forall v \in H_\varepsilon^* \rho(v \cdot \varepsilon) = \rho(v) \;\&\; \forall u \in R(H)\rho(u) = u.$$

• *удаление всех пустых символов* μ: $H_\varepsilon^* \to H^*$ определяется условием:

$$\forall v, w \in H_\varepsilon^* \mu(v \cdot \varepsilon \cdot \mu) = \mu(v \cdot w) \;\&\; \forall u \in H^* \mu(u) = u.$$

Определим для последовательности $x \in H_\varepsilon^*$:

• *E-вложение* получается из x заменой некоторых символов пустыми символами;

• *L-вложение* получается из *E*-вложения удалением префикса пустых символов;

• *R-вложение* получается из *E*-вложения удалением постфикса пустых символов;

• *O-вложение* получается из *E*-вложения удалением префикса и постфикса пустых символов, или из *L*-вложения удалением постфикса пустых символов, или из *R*-вложения удалением префикса пустых символов;

• *A-вложение* получается из *E*-, *L*-, *R*- или *O*-вложения удалением пустых символов.

Для $x \in H^*$ понятие *A*-вложения совпадает с понятием подпоследовательности, а *E*-вложение v соответствует понятию вхождения подпоследовательности $\mu(v)$ в x.

Обозначим множества вложений в последовательность $x \in H^*$:

• множество *E*-вложений в x:

$$E(x) = \{u \in H_\varepsilon^* : |u| = |x| \;\&\; \forall i \in 1..|u| u_i = x_i \vee u_i = \varepsilon\};$$

• множество *L*-вложений в x:

$$L(x) = \lambda E(x);$$

• множество *R*-вложений в x:

$$R(x) = \rho E(x);$$

• множество *O*-вложений в x:

$$O(x) = \lambda r E(x) = \lambda R(x) = \rho L(x);$$

• множество *A*-вложений в x:

$$A(x) = \mu E(x) = \mu L(x) = \mu R(x) = \mu O(x).$$

Для последовательности x в алфавите H_ε^* введем понятие μ-*длины* x как число непустых символов в x, очевидно, равное $|\mu(x)|$. Для $x \in H^*$ μ-длина совпадает с длиной последовательности.

Обозначим для последовательностей $x \in H^*$:

• множество *E*-вложений в x *L*-вложения $u \in L(x)$: $l(u, x) = \{v \in E(x) : \lambda(v) = u\}$;

• множество *E*-вложений в x *R*-вложения $u \in R(x)$: $r(u, x) = \{v \in E(x) : \rho(v) = u\}$;

• множество *E*-вложений в x *O*-вложения $u \in O(x)$: $o(u, x) = \{v \in E(x) : \lambda\rho(v) = u\}$;

• множество *E*-вложений в x *A*-вложения $u \in A(x)$: $a(u, x) = \{v \in E(x) : \mu(v) = u\}$;

Обозначим для последовательностей $x \in H^*$ и $y \in H^*$ множества пар *E*-вложений:

• множество пар *E*-вложений общих *L*-вложений

$$L(x, y) = \cup\{l(u, x) \times l(u, y) : u \in L(x) \cap L(y)\};$$

• множество пар *E*-вложений общих *R*-вложений

$$R(x, y) = \cup\{r(u, x) \times r(u, y) : u \in R(x) \cap R(y)\};$$

• множество пар *E*-вложений общих *O*-вложений

$$O(x, y) = \cup\{o(u, x) \times o(u, y) : u \in O(x) \cap O(y)\};$$

• множество пар *E*-вложений общих *A*-вложений

$$A(x, y) = \cup\{a(u, x) \times a(u, y) : u \in A(x) \cap A(y)\}.$$

Обозначим для последовательностей $x \in H^*$ и $y \in H^*$ наибольшую μ-длину общего вложения:

• для *L*-вложений: $lcL(x, y) = max\{\mu(u) : u \in L(x) \cap L(y)\}$;

• для *R*-вложений: $lcR(x, y) = max\{\mu(u) : u \in R(x) \cap R(y)\}$;

• для *O*-вложений: $lcO(x, y) = max\{\mu(u) : u \in O(x) \cap O(y)\}$;

• для *A*-вложений: $lcA(x, y) = max\{|u| : u \in A(x) \cap A(y)\}$;

Наибольшую μ-длину общего вложения естественно рассматривать как еще одну функцию “похожести” последовательностей x и y: $lcI(x, y)$ для $I \in \{L, R, O, A\}$. По этому критерию последовательность x больше всего “похожа” на саму себя, поскольку является наибольшим *I*-вложением в себя: $lcI(x, x) = max\{\mu(u) : u \in I(x)\} = |x| \geq max\{\mu(u) : u \in I(x) \cap I(y)\} = lcI(x, y)$.

Нас будут интересовать функции от последовательностей $x \in H^*$ и $y \in H^*$, задаваемые табл. 1.

Заметим, что для $I \in \{L, R, O, A\}$ и $j \in 0..4$ $I_j(x, y) = I_j(y, x)$. Если $j \neq 1$, то $I_j(x, ()) = I_j((), y) = 1$; $I_1(x, ()) = I_1((), y) = 0$.

Обозначим для непустой последовательности $x \in H^*$:

• самое левое *E*-вложение в x *L*-вложения $u \in L(x)$: $l_l(u, x) = v$, если $v \in l(u, x)$ и $\forall w \in l(u, x)\backslash\{v\}$ $w(min\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$;

• самое правое *E*-вложение в x *L*-вложения $u \in O(x)$: $l_r(u, x) = v$, если $v \in l(u, x)$ и $\forall w \in l(u, x)\backslash\{v\}$ $w(max\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$.

• самое левое *E*-вложение в x *R*-вложения $u \in R(x)$: $r_l(u, x) = v$, если $v \in r(u, x)$ и $\forall w \in r(u, x)\backslash\{v\}$ $w(min\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$;

• самое правое *E*-вложение в x *R*-вложения $u \in O(x)$: $r_r(u, x) = v$, если $v \in r(u, x)$ и $\forall w \in r(u, x)\backslash\{v\}$ $w(max\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$.

Таблица 1. Функции для общих вложений двух последовательностей x и y

Функция\ Вложение	Число общих вложений	Сумма μ-длин общих вложений	Сумма минимумов чисел E-вложений общих вложений	Сумма произведений чисел E-вложений общих вложений	Наибольшая μ-длина общего вложения
	0	1	2	3	4
L	$L_0(x, y) = \|L(x) \cap L(y)\|$	$L_1(x, y) = \Sigma\{\|\mu(u)\| : u \in L(x) \cap L(y)\}$	$L_2(x, y) = \Sigma\{min\{\|l(u, x)\|, \|l(u, y)\|\} : u \in L(x) \cap L(y)\}$	$L_3(x, y) = \|L(x, y)\|$	$L_4(x, y) = lcL(x, y)$
R	$R_0(x, y) = \|R(x) \cap R(y)\|$	$R_1(x, y) = \Sigma\{\|\mu(u)\| : u \in R(x) \cap R(y)\}$	$R_2(x, y) = \Sigma\{min\{\|r(u, x)\|, \|r(u, y)\|\} : u \in R(x) \cap R(y)\}$	$R_3(x, y) = \|R(x, y)\|$	$R_4(x, y) = lcR(x, y)$
O	$O_0(x, y) = \|O(x) \cap O(y)\|$	$O_1(x, y) = \Sigma\{\|\mu(u)\| : u \in O(x) \cap O(y)\}$	$O_2(x, y) = \Sigma\{min\{\|o(u, x)\|, \|o(u, y)\|\} : u \in O(x) \cap O(y)\}$	$O_3(x, y) = \|O(x, y)\|$	$O_4(x, y) = lcO(x, y)$
A	$A_0(x, y) = \|A(x) \cap A(y)\|$	$A_1(x, y) = \Sigma\{\|u\| : u \in A(x) \cap A(y)\}$	$A_2(x, y) = \Sigma\{min\{\|a(u, x)\|, \|a(u, y)\|\} : u \in A(x) \cap A(y)\}$	$A_3(x, y) = \|A(x, y)\|$	$A_4(x, y) = lcA(x, y)$

• самое левое E-вложение в x O-вложения $u \in O(x)$: $o_l(u, x) = v$, если $v \in o(u, x)$ и $\forall\, w \in o(u, x)\backslash\{v\}$ $w(min\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$;

• самое правое E-вложение в x O-вложения $u \in O(x)$: $o_r(u, x) = v$, если $v \in o(u, x)$ и $\forall\, w \in o(u, x)\backslash\{v\}$ $w(max\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$.

• самое левое E-вложение в x A-вложения $u \in A(x)$: $a_l(u, x) = v$, если $v \in a(u, x)$ и $\forall\, w \in a(u, x)\backslash\{v\}$ $w(min\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$;

• самое правое E-вложение в x A-вложения $u \in A(x)$: $a_r(u, x) = v$, если $v \in a(u, x)$ и $\forall\, w \in a(u, x)\backslash\{v\}$ $w(max\{i \in 1..|x| : w_i \neq v_i\}) = \varepsilon$.

В литературе самые левые (*left-most*) E-вложения общих вложений называют также (для подпоследовательностей) каноническими (*canonical*) [3].

Обозначим для непустых последовательностей $x \in H^*$ и $y \in H^*$ множества пар E-вложений:

• множество пар самых левых E-вложений общих L-вложений

$$L_l(x, y) = \{(l_l(u, x), l_l(u, y)) : u \in L(x) \cap L(y)\};$$

• множество пар самых правых E-вложений общих L-вложений

$$L_r(x, y) = \{(l_r(u, x), l_r(u, y)) : u \in L(x) \cap L(y)\};$$

• множество пар самых левых E-вложений общих R-вложений

$$R_l(x, y) = \{(r_l(u, x), r_l(u, y)) : u \in R(x) \cap R(y)\};$$

• множество пар самых правых E-вложений общих R-вложений

$$R_r(x, y) = \{(r_r(u, x), r_r(u, y)) : u \in R(x) \cap R(y)\};$$

• множество пар самых левых E-вложений общих O-вложений

$$O_l(x, y) = \{(o_l(u, x), o_l(u, y)) : u \in O(x) \cap O(y)\};$$

• множество пар самых правых E-вложений общих O-вложений

$$O_r(x, y) = \{(o_r(u, x), o_r(u, y)) : u \in O(x) \cap O(y)\};$$

• множество пар самых левых E-вложений общих A-вложений

$$A_l(x, y) = \{(a_l(u, x), a_l(u, y)) : u \in A(x) \cap A(y)\}.$$

• множество пар самых правых E-вложений общих A-вложений

$$A_r(x, y) = \{(a_r(u, x), a_r(u, y)) : u \in A(x) \cap A(y)\};$$

Заметим, что $|O_l(x, y)| = |O_r(x, y)| = |O(x) \cap O(y)|$ и $|A_l(x, y)| = |A_r(x, y)| = |A(x) \cap A(y)|$.

Далее через x и y будем обозначать две непустые последовательности в алфавите H с длинами $m = |x|$, $n = |y|$. Будем считать, что $m \leq n$.

3. ЗАМЕНА НЕОБЩИХ СИМВОЛОВ ПУСТЫМ СИМВОЛОМ

Общей оптимизацией для вычисления всех функций для всех вложений является замена необщих символов пустым символом: для $i \in 1..m$ определим $x_i^\wedge = x_i$, если $x_i \in \mathbf{Im}\, y$, $x_i^\wedge = \varepsilon_i$, если $x_i \notin \mathbf{Im}\, y$. Аналогично определяется $y^\wedge$. Эта замена выполняется за время $\mathbf{O}(mn)$. В случае A-вложений вместо замены необщего символа пустым символом

можно просто удалять необщий символ. Описанные ниже алгоритмы можно применять после выполнения этой замены (удаления для A-вложений).

4. A-ВЛОЖЕНИЯ (ПОДПОСЛЕДОВАТЕЛЬНОСТИ)

4.1. Число общих A-вложений

Здесь мы докажем теорему, аналогичную лемме 6 в [3], но дадим свое доказательство, поскольку мы используем другие определения и обозначения.

Для непустой последовательности $z \in H^*$ и символа $h \in H$ обозначим максимальный индекс, по которому в последовательности z находится символ h, или 0, если h не входит в z:

$p(z, h) = max\{i \in 1..|z|: z_i = h\}$, если $h \in \mathbf{Im}\ z$; $p(z, h) = 0$, если $h \notin \mathbf{Im}\ z$.

Обозначим: $k = p(x[m-1], x_m\}$, $l = p(y, x_m\}$.

Теорема 1.

$A_0(x, y) = A_0(x[m-1], y)$, если $x_m \notin \mathbf{Im}\ y$;

$A_0(x, y) = A_0(x[m-1], y) + A_0(x[m-1], y[l-1])$, если $x_m \in \mathbf{Im}\ y$ и $x_m \notin \mathbf{Im}\ x[m-1]$;

$A_0(x, y) = A_0(x[m-1], y) + A_0(x[m-1], y[l-1]) - A_0(x[k-1], y[l-1])$, если $x_m \in \mathbf{Im}\ y$ и $x_m \in \mathbf{Im}\ x[m-1]$.

Доказательство.

Рассматриваем множества пар $(e_r(u, x), e_r(u, y))$ самых правых E-вложений общих A-вложений u последовательностей x и y. Обозначим пары последовательностей длины m и n:

$E = A_r(x, y)$;

$E_m = A_r(x[m-1], y)(\varepsilon, ())$;

$E_{ml} = A_r(x[m-1], y[l-1])(x_m, x_m\varepsilon^{n-l})$, если $x_m \in \mathbf{Im}\ y$, иначе $E_{ml} = \varnothing$;

$E_{kl} = A_r(x[k-1], y[l-1])(x_m\varepsilon^{m-k-1}x_m, x_m\varepsilon^{n-l})$, если $x_m \in \mathbf{Im}\ y$ и $x_m \in \mathbf{Im}\ x[m-1]$, иначе $E_{kl} = \varnothing$.

Поскольку для любых x', y' имеет место $A_0(x', y') = |A(x') \cap A(y')| = |A_r(x', y')|$, утверждение теоремы можно переписать в виде:

$|E| = |E_m|$, если $x_m \notin \mathbf{Im}y$;

$|E| = |E_m| + |E_{ml}|$, если $x_m \in \mathbf{Im}y$ и $x_m \notin \mathbf{Im}x[m-1]$;

$|E| = |E_m| + |E_{ml}| - |E_{kl}|$, если $x_m \in \mathbf{Im}y$ и $x_m \in \mathbf{Im}x[m-1]$.

Поскольку E-вложения из множества E_m имеют вид $(...\varepsilon, ...)$, а множества E_{ml} и E_{kl} либо пусты, либо E-вложения из них имеют вид $(...x_m, ...)$, имеем $E_m \cap E_{ml} = E_m \cap E_{kl} = \varnothing$. Поскольку $m - 1 \geq k - 1$, имеем $E_{ml} \supseteq E_{kl}$.

Рассмотрим случай $x_m \notin \mathbf{Im}\ y$. Поскольку $x_m \notin \mathbf{Im}\ y$, переход от последовательности $x[m-1]$ к последовательности $x = x[m-1]x_m$ не добавляет новых общих A-вложений. Поэтому $E = E_m$. Отсюда $|E| = |E_m|$, что и требовалось доказать в этом случае.

Рассмотрим случай $x_m \in \mathbf{Im}\ y$ и $x_m \notin \mathbf{Im}\ x[m-1]$. Поскольку $x_m \in \mathbf{Im}\ y$, переход от последовательности $x[m-1]$ к последовательности $x = x[m-1]x_m$ может добавлять новые общие A-вложения, эти общие A-вложения должны заканчиваться на x_m, но этих общих A-вложений раньше, в $x[m-1]$ и y, не было, поскольку $x_m \notin \mathbf{Im}\ x[m-1]$. Заметим, что для такого нового общего A-вложения u в его самых правых E-вложениях в x и y последний непустой символ равен x_m по индексам m и l, соответственно, т.е. $e_r(u, x)_m = e_r(u, y)_l = x_m$. Имеем $E = E_m \cup E_{ml}$. Поскольку $E_m \cap E_{ml} = \varnothing$, имеем $|E| = |E_m| + |E_{ml}|$, что и требовалось доказать в этом случае.

Теперь рассмотрим случай $x_m \in \mathbf{Im}y$ и $x_m \in \mathbf{Im}\ x[m-1]$. Поскольку $x_m \in \mathbf{Im}y$, переход от последовательности $x[m-1]$ к последовательности $x = x[m-1]x_m$ может добавлять новые общие A-вложения u, но они новые только в том случае, когда таких A-вложений не было раньше в $x[m-1]$ и y. Эти новые общие A-вложения должны заканчиваться на x_m. Заметим, что если такое общее A-вложение u было раньше, то в его самых правых E-вложениях в $x[m-1]$ и y последний непустой символ равен x_m по индексам k и l, соответственно, т.е. $e_r(u, x[m-1])_k = e_r(u, y)_l = x_m$. В любом случае в самых правых E-вложениях u в x и y последний непустой символ равен x_m по индексам m и l, соответственно, т.е. $e_r(u, x)_m = e_r(u, y)_l = x_m$. Имеем $E = E_m \cup (E_{ml} \backslash E_{kl})$. Поскольку $E_m \cap E_{ml} = E_m \cap E_{kl} = \varnothing$, имеем $E_m \cap (E_{ml} \backslash E_{kl}) = \varnothing$ и поэтому $|E| = |E_m| + |E_{ml} \backslash E_{kl}|$. Поскольку $E_{ml} \supseteq E_{kl}$, имеем $|E_{ml} \backslash E_{kl}| = |E_{ml}| - |E_{kl}|$. Поэтому $|E| = |E_m| + |E_{ml}| - |E_{kl}|$, что и требовалось доказать в этом случае.

□

Теорема 1 определяет алгоритм вычисления $A_0(x, y)$. Число шагов алгоритма равно $\mathbf{O}(mn)$, что определяется числом функций вида $A_0(x[m-i], y[n-j])$, где $i \in 0..m$ и $j \in 0..n$, при условии, что каждая функция вычисляется не более одного раза (после чего ее значение сохраняется). На каждом шаге проверка условий $x_{m-i} \in \mathbf{Im}\ y[n-j]$ и $x_{m-i} \in \mathbf{Im}\ x[m-i-1]$ имеет сложность $\mathbf{O}(m+n)$, а остальные вычисления имеют сложность $\mathbf{O}(1)$. Сложность алгоритма равна $\mathbf{O}(m^2n + mn^2)$, что для $m \leq n$ равно $\mathbf{O}(mn^2)$.

4.2. Сумма длин общих A-вложений

Теорема 2.

A1(x, y) = A1(x[m − 1], y), если xm ∉ Im y;

$A_1(x, y) = A_1(x[m-1], y) + A_1(x[m-1], y[l-1]) + A_0(x[m-1], y[l-1])$, если $x_m \in \mathbf{Im}\ y$ и $x_m \notin \mathbf{Im}\ x[m-1]$;

$A_1(x, y) = A_1(x[m-1], y) + A_1(x[m-1], y[l-1]) + A_0(x[m-1], y[l-1]) - A_1(x[k-1], y[l-1]) -$

$A_0(x[k-1], y[l-1])$, если $x_m \in \mathbf{Im}\, y$ и $x_m \in \mathbf{Im}\, x[m-1]$.

Доказательство.

Доказательство аналогично доказательству теоремы 1. Используем те же обозначения:

$E = A_r(x, y)$;

$E_m = A_r(x[m-1], y)(\varepsilon, ())$;

$E_{ml} = A_r(x[m-1], y[l-1])(x_m, x_m\varepsilon^{n-l})$, если $x_m \in \mathbf{Im}\, y$, иначе $E_{ml} = \varnothing$;

$E_{kl} = A_r(x[k-1], y[l-1])(x_m\varepsilon^{m-k-1}x_m, x_m\varepsilon^{n-l})$, если $x_m \in \mathbf{Im}\, y$ и $x_m \in \mathbf{Im}\, x[m-1]$, иначе $E_{kl} = \varnothing$.

Имеем $E_m \cap E_{ml} = E_m \cap E_{kl} = \varnothing$ и $E_{ml} \supseteq E_{kl}$.

Также имеем $A_0(x[m-1], y[l-1]) = |E_{ml}|$ и $|A_0(x[k-1], y[l-1])| = |E_{kl}|$.

Обозначим:

$S = \Sigma\,\{|u| : u \in A(x) \cap A(y)\ \&\ (e_r(u, x), e_r(u, y)) \in E\}$,

$S_m = \Sigma\,\{|u| : u \in A(x) \cap A(y)\ \&\ (e_r(u, x), e_r(u, y)) \in E_m\}$,

$S_{ml} = \Sigma\,\{|u| : u \in A(x) \cap A(y)\ \&\ (e_r(u, x), e_r(u, y)) \in E_{ml}\}$, если $x_m \in \mathbf{Im}\, y$, иначе $E_{ml} = \varnothing$,

$S_{kl} = \Sigma\,\{|u| : u \in A(x) \cap A(y)\ \&\ (e_r(u, x), e_r(u, y)) \in E_{kl}\}$, если $x_m \in \mathbf{Im}\, y$ и $x_m \in \mathbf{Im}\, x[m-1]$, иначе $E_{kl} = \varnothing$.

Длина общего A-вложения u равна числу непустых символов в каждом его E-вложении, в том числе, в самом правом E-вложении. Поэтому в этих обозначениях утверждение теоремы можно переписать в виде:

$S = S_m$, если $x_m \notin \mathbf{Im}\, y$;

$S = S_m + S_{ml} + |E_{ml}|$, если $x_m \in \mathbf{Im}\, y$ и $x_m \notin \mathbf{Im}\, x[m-1]$;

$S = S_m + S_{ml} + |E_{ml}| - S_{kl} - |E_{kl}|$, если $x_m \in \mathbf{Im}\, y$ и $x_m \in \mathbf{Im}\, x[m-1]$.

При переходе от последовательности $x[m-1]$ к последовательности $x = x[m-1]x_m$ число непустых символов в самом правом E-вложении в x общего A-вложения u не меняется, если в нем по индексу m оказывается пустой символ $e_r(u, x)_m = \varepsilon$, и увеличивается на 1 в противном случае, когда $e_r(u, x)_m = x_m$.

Это влечет следующие утверждения.

В случае $x_m \notin \mathbf{Im}\, y$ имеем $E = E_m$, для каждого $(u, v) \in E_m$ имеет место $u_m = \varepsilon$, поэтому $S = S_m$, что и требовалось доказать в этом случае.

В случае $x_m \in \mathbf{Im}\, y$ и $x_m \notin \mathbf{Im}\, x[m-1]$ имеем $E = E_m \cup E_{ml}$ и $E_m \cap E_{ml} = \varnothing$, для каждого $(u, v) \in E_m$ имеет место $u_m = \varepsilon$, для каждого $(u, v) \in E_{ml}$ имеет место $u_m = x_m$, поэтому $S = S_m + (S_{ml} + |E_{ml}|)$, что и требовалось доказать в этом случае.

В случае $x_m \in \mathbf{Im}\, y$ и $x_m \in \mathbf{Im}\, x[m-1]$ имеем $E = E_m \cup (E_{ml} \backslash E_{kl})$ и $E_m \cap E_{ml} = E_m \cap E_{kl} = \varnothing$, для каждого $(u, v) \in E_m$ имеет место $u_m = \varepsilon$, для каждого $(u, v) \in E_{ml}$ имеет место $u_m = x_m$, для каждого $(u, v) \in E_{kl}$ имеет место $u_m = x_m$, поэтому $S = S_m + (S_{ml} + |E_{ml}|) - (S_{kl} + |E_{kl}|)$, что и требовалось доказать в этом случае.

□

Теорема 2 определяет алгоритм вычисления $A_1(x, y)$, сложность которого по порядку, очевидно, не превышает сложности алгоритма вычисления $A_0(x, y)$, т.е. равна $\mathbf{O}(m^2n + mn^2)$, что для $m \le n$ равно $\mathbf{O}(mn^2)$.

4.3. Сумма минимумов чисел E-вложений общих A-вложений

Для функции $A_2(x, y)$ мы не знаем хорошего (отличного от полного перебора) алгоритма. Единственная полезная оптимизация – это предварительное удаление необщих символов.

4.4. Сумма произведений чисел E-вложений общих A-вложений

Здесь мы докажем теорему, аналогичную теореме 2 в [3] с тем отличием, что мы учитываем пустую подпоследовательность, а в теореме 2 в [3] она не учитывается. Также мы дадим свое доказательство, поскольку мы используем другие определения и обозначения.

Теорема 3. $A_3(x, y) = A_3(x[m-1], y) + A_3(x, y[n-1]) - A_3(x[m-1], y[n-1])$, если $x_m \ne y_n$;

$A_3(x, y) = A_3(x[m-1], y) + A_3(x, y[n-1])$, если $x_m = y_n$.

Доказательство.

Обозначим следующие множества пар E-вложений общих A-вложений:

$E = A(x, y)$,

$E_{00} = A(x[m-1], y[n-1])(\varepsilon, \varepsilon)$ – пара последних элементов $(\varepsilon, \varepsilon)$,

$E_{01} = (A(x[m-1], y])(\varepsilon, ()))\backslash E_{00}$ – пара последних элементов (ε, y_n),

$E_{10} = (A(x, y[n-1])((), \varepsilon))\backslash E_{00}$ – пара последних элементов (x_m, ε),

$E_{11} = E\backslash(E_{00} \cup E_{01} \cup E_{10})$ – пара последних элементов (x_m, y_n), поскольку $E_{00} \cup E_{01} \cup E_{10}$ содержат все пары E-вложений общих A-вложений, в которых пара последних элементов содержит ε.

Очевидно, $E = E_{00} \cup E_{01} \cup E_{10} \cup E_{11}$. Пары последних элементов пар E-вложений из разных множеств E_{00}, E_{01}, E_{10}, E_{11} разные, поэтому эти множества попарно не пересекаются. Поэтому $|E| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{11}|$, $|E_{00} \cup E_{01}| = |E_{00}| + |E_{01}|$, $|E_{00} \cup E_{10}| = |E_{00}| + |E_{10}|$.

Поскольку $A_3(x, y) = |A(x, y)|$, в этих обозначениях утверждение теоремы имеет вид

$|E| = (|E_{00}| + |E_{01}|) + (|E_{00}| + |E_{10}|) - |E_{00}| = |E_{00}| + |E_{01}| + |E_{10}|$, если $x_m \neq y_n$,

$|E| = (|E_{00}| + |E_{01}|) + (|E_{00}| + |E_{10}|) = 2|E_{00}| + |E_{01}| + |E_{10}|$, если $x_m = y_n$.

Рассмотрим случай $x_m \neq y_n$. В этом случае $E_{11} = \varnothing$, что влечет $|E_{11}| = 0$ и $|E| = |E_{00}| + |E_{01}| + |E_{10}|$, что и требовалось доказать в этом случае.

Рассмотрим случай $x_m = y_n = h$. Каждая пара E-вложений из E_{11} имеет вид $(u\varepsilon^{m-|u|-1}h, v\varepsilon^{n-|v|-1}h)$, где $(u\varepsilon^{m-|u|-1}\varepsilon, v\varepsilon^{n-|v|-1}\varepsilon) \in E_{00}$ (последовательности u и v могут быть обе пустыми). Будем говорить, что пара $(u\varepsilon^{m-|u|-1}h, v\varepsilon^{n-|v|-1}h)$ соответствует паре $(u\varepsilon^{m-|u|-1}\varepsilon, v\varepsilon^{n-|v|-1}\varepsilon)$. Это соответствие, очевидно, является биекцией множеств E_{11} и E_{00}. Тем самым, $|E_{11}| = |E_{00}|$. Поэтому $|E| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{11}| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{00}| = 2|E_{00}| + |E_{01}| + |E_{10}|$, что и требовалось доказать в этом случае.

□

Теорема 3 определяет алгоритм вычисления $A_3(x, y)$. Число шагов алгоритма равно $\mathbf{O}(mn)$, что определяется числом функций вида $A_3(x[m - i], y[n - j])$, где $i \in 0..m$ и $j \in 0..n$, при условии, что каждая функция вычисляется не более одного раза (после чего ее значение сохраняется). На каждом шаге вычисления имеют сложность $\mathbf{O}(1)$. Тем самым сложность алгоритма равна $\mathbf{O}(mn)$.

4.5. Функция похожести на основе наибольшей длины общего A-вложения

Задача вычисления длины наибольшей общей подпоследовательности (*longest common subsequence*, *lcs*) хорошо известна [1]. Простейший алгоритм сложности $\mathbf{O}(mn)$ основан на следующих соотношениях:

$lcA(x, y) = lcA(x[m - 1], y[n - 1]) + 1$, если $x_m = y_n$;

$lcA(x, y) = max\{lcA(x, y[n - 1]), lcA(x[m - 1], y)\}$, если $x_m \neq y_n$.

Тем самым, функция $A_4(x, y) = lcA(x, y)$ вычисляется за время $\mathbf{O}(mn)$.

5. L-ВЛОЖЕНИЯ

В отличие от A- и O-вложений L-вложению $u \in L(x)$ соответствует только одно E-вложение v такое, что $\lambda(v) = u$. Общему L-вложению $u \in L(x) \cap \cap L(y)$ соответствует пара E-вложений в x и y, которые могут отличаться только префиксом пустых символов. Поэтому множество $l(u, x)$ является синглетоном, $\{l_l(u, x)\} = \{l_r(u, x)\} = l(u, x)$, $L(x, y) = L_l(x, y) = L_r(x, y)$.

Обозначим через $\gamma(x, y)$ последовательность z длиной m (напомним, что $|x| = m \le n = |y|$), совпадающую с x и y по тем позициям, считая справа налево, по которым совпадают x и y, и содержащую пустой символ по остальным позициям: $|z| = m$ и $\forall i \in 1..m\ (x_{m+1-i} = y_{n+1-i} \Rightarrow z_{m+1-i} = x_{m+1-i})\ \&\ (x_{m+1-i} \neq y_{n+1-i} \Rightarrow z_{m+1-i} = \varepsilon)$. Функция $\gamma(x, y)$ устанавливает позиционное соответствие совпадающих символов x и y при счете позиций справа налево.

5.1. Число общих L-вложений

Теорема 4. $L_0(x, y) = L_0(x[m - 1], y[n - 1]))$, если $x_m \neq y_n$,

$L_0(x, y) = 2L_0(x[m - 1], y[n - 1])$, если $x_m = y_n$.

Доказательство.

Обозначим $L = L(x) \cap L(y)$ и $L_{-1} = L(x[m - 1]) \cap \cap L(y[n - 1])$. Если $x_m \neq y_n$, то L-вложению $u \in L_{-1}$ соответствует одно L-вложение $u\varepsilon \in L$. Тем самым, $L_0(x, y) = |L| = |L_{-1}| = L_0(x[m - 1], y[n - 1]))$. Если $x_m = y_n$, то L-вложению $u \in L_{-1}$ соответствуют два L-вложения $u\varepsilon \in L$ и $ux_m \in L$. Тем самым, $L_0(x, y) = |L| = 2|L_{-1}| = 2L_0(x[m - 1], y[n - 1]))$.

□

Теорема 4 определяет алгоритм вычисления $L_0(x, y)$. Для $m \le n$ число шагов алгоритма равно $\mathbf{O}(m)$, что определяется числом функций вида $L_0(x[m - i], y[n - i])$, где $i \in 0..m$. На каждом шаге вычисления имеют сложность $\mathbf{O}(1)$. Тем самым сложность алгоритма равна $\mathbf{O}(m)$.

Теорема 5. $L_0(x, y) = 2^{|\mu\gamma(x, y)|}$.

Доказательство.

Из теоремы 4 следует, что при добавлении к обеим последовательностям справа по одному символу число общих L-вложений увеличивается вдвое, если добавляются равные символы, и не меняется в противном случае. Поскольку $|L(x) \cap \cap L(())| = |L(()) \cap L(y)| = |\varepsilon| = 1$, а $|\mu\gamma(x, y)|$ равно числу совпадающих символов в одинаковых позициях x и y при отсчете справа налево, имеем $L_0(x, y) = |L(x) \cap L(y)| = 2^{|\mu\gamma(x, y)|}$, что и требовалось доказать.

□

Теорема 5 определяет алгоритм вычисления $L_0(x, y)$. Сложность алгоритма равна сложности вычисления $\mu\gamma(x, y)$, которая, очевидно, для $m \le n$ равна $\mathbf{O}(m)$, плюс сложность возведения числа два в степень $|\mu\gamma(x, y)|$, которая тоже равна $\mathbf{O}(m)$. Тем самым сложность алгоритма равна $\mathbf{O}(m)$.

5.2. Сумма μ-длин общих L-вложений

Теорема 6. $L_1(x, y) = L_1(x[m - 1], y[n - 1]))$, если $x_m \neq y_n$,

$L_1(x, y) = 2L_1(x[m - 1], y[n - 1] + L_0(x[m - 1], y[n - 1])$, если $x_m = y_n$.

Доказательство.

Обозначим $L = L(x) \cap L(y)$ и $L_{-1} = L(x[m-1]) \cap \cap L(y[n-1])$. Если $x_m \neq y_n$, то L-вложению $u \in L_{-1}$ соответствует одно L-вложение $u\varepsilon \in L$ той же μ-длины. Тем самым, $L_1(x, y) = \Sigma\{|\mu(u)| : u \in L\} = \Sigma\{|\mu(u)| : u \in L_{-1}\} = L_1(x[m-1], y[n-1]))$. Если $x_m = y_n$, то L-вложению $u \in L_{-1}$ соответствуют два L-вложения $u\varepsilon \in L$ той же μ-длины и $ux_m \in L$ с μ-длиной на 1 большей. Тем самым, $L_1(x, y) = \Sigma\{|\mu(u)| : u \in L\} = \Sigma\{|\mu(u)| : u \in L_{-1}\} + (\Sigma\{|\mu(u)| : u \in L_{-1}\} + L_0(x[m-1], y[n-1])) = 2\Sigma\{|\mu(u)| : u \in L_{-1}\} + L_0(x[m-1], y[n-1]) = 2L_1(x[m-1], y[n-1] + L_0(x[m-1], y[n-1])$.

□

Теорема 6 определяет алгоритм вычисления $L_1(x, y)$, сложность которого по порядку, очевидно, не превышает сложности алгоритма вычисления $L_0(x, y)$, т.е. равна $\mathbf{O}(m)$.

Теорема 7. $L_1(x, y) = 1{*}C_l^1 + 2{*}C_l^2 + \ldots + l{*}C_l^1$ (A001787), где $l = |\mu\gamma(x, y)|$.

Доказательство.

Утверждение теоремы непосредственно следует из того факта, что число общих L-вложений μ-длины i равно $C^i_{|\mu\gamma(x,y)|}$.

□

Теорема 7 определяет алгоритм вычисления $L_1(x, y)$. Сложность алгоритма равна сложности вычисления $\mu\gamma(x, y)$, которая, очевидно, для $m \le n$ равна $\mathbf{O}(m)$, плюс сложность вычисления факториалов $i!$ для $i \in 0..l$, равная $\mathbf{O}(m)$, плюс сложность суммирования l чисел, равная $\mathbf{O}(m)$. Тем самым сложность алгоритма равна $\mathbf{O}(m)$.

5.3. Сумма минимумов и сумма произведений чисел E-вложений общих L-вложений

Теорема 8. $L_2(x, y) = L_3(x, y) = L_0(x, y)$.

Доказательство.

Утверждение теоремы непосредственно следует из того факта, что L-вложению $u \in L(x)$ соответствует ровно одно E-вложение v такое, что $\lambda(v) = u$: $L_2(x, y) = \Sigma\{min\{|l(u, x)|, |l(u, y)|\} : u \in L(x) \cap \cap L(y)\} = \Sigma\{min\{1, 1\} : u \in L(x) \cap L(y)\} = |L(x) \cap \cap L(y)| = L_0(x, y)$; $L_3(x, y) = |L(x, y)| = |\cup\{l(u, x) \times \times l(u, y) : u \in L(x) \cap L(y)\}| = |L(x) \cap L(y)| = L_0(x, y)$.

□

Теорема 8 определяет алгоритм вычисления $L_2(x, y)$ и $L_3(x, y)$ той же сложности, что алгоритм вычисления $L_0(x, y)$, т.е. для $m \le n$ сложности $\mathbf{O}(m)$.

5.4. Функция похожести на основе наибольшей длины общего L-вложения

Теорема 9.

$lcL(x, y) = lcL(x[m-1], y[n-1]) + 1$, если $x_m = y_n$;

$lcL(x, y) = lcL(x[m-1], y[n-1])$, если $x_m \neq y_n$.

Доказательство. Функция $\gamma(x, y)$ устанавливает позиционное соответствие совпадающих символов x и y при счете позиций справа налево. Поэтому $lcL(x, y) = |\mu\gamma(x, y)|$. Отсюда непосредственно следует утверждение теоремы.

□

Теорема 9 определяет алгоритм вычисления функции $L_4(x, y) = lcL(x, y)$ сложности $\mathbf{O}(m)$ для $m \le n$.

6. *R*-ВЛОЖЕНИЯ

Аналогично L-вложению R-вложению $u \in R(x)$ соответствует только одно E-вложение v такое, что $\rho(v) = u$. Общему R-вложению $u \in R(x) \cap R(y)$ соответствует пара E-вложений в x и y, отличающихся только постфиксом пустых символов. Поэтому множество $r(u, x)$ является синглетоном, $\{r_l(u, x)\} = \{r_r(u, x)\} = r(u, x)$, $R(x, y) = R_l(x, y) = R_r(x, y)$.

Обозначим через $\delta(x, y)$ последовательность z длиной m (напомним, что $|x| = m \le n = |y|$), совпадающую с x и y по тем позициям, считая слева направо, по которым совпадают x и y, и содержащую пустой символ по остальным позициям: $|z| = m$ и $\forall i \in 1..m\ (x_i = y_i \Rightarrow z_i = x_i)\ \&\ (x_i \neq y_i \Rightarrow z_i = \varepsilon)$. Функция $\delta(x, y)$ устанавливает позиционное соответствие совпадающих символов x и y при счете позиций слева направо.

6.1. Число общих R-вложений

Теорема 10. $R_0(x, y) = R_0(x[2..m], y[2..n]))$, если $x_1 \neq y_1$,

$R_0(x, y) = 2R_0(x[2..m], y[2..n])$, если $x_1 = y_1$.

Доказательство.

Обозначим $R = R(x) \cap R(y)$ и $R_{-1} = R(x[2..m]) \cap \cap R(y[2..n])$. Если $x_1 \neq y_1$, то R-вложению $u \in R_{-1}$ соответствует одно R-вложение $\varepsilon u \in R$. Тем самым, $R_0(x, y) = |R| = |R_{-1}| = R_0(x[2..m], y[2..n]))$. Если $x_1 = y_1$, то R-вложению $u \in R_{-1}$ соответствуют два R-вложения $\varepsilon u \in R$ и $x_1 u$. Тем самым, $R_0(x, y) = = |R| = 2|R_{-1}| = 2R_0(x[2..m], y[2..n]))$.

□

Теорема 10 определяет алгоритм вычисления $R_0(x, y)$. Для $m \le n$ число шагов алгоритма равно $\mathbf{O}(m)$, что определяется числом функций вида $R_0(x[i..m], y[i..n])$, где $i \in 0..m$. На каждом шаге проверка вычисления имеет сложность $\mathbf{O}(1)$. Тем самым сложность алгоритма равна $\mathbf{O}(m)$.

Теорема 11. $R_0(x, y) = 2^{|\mu\delta(x, y)|}$.

Доказательство.

Из теоремы 10 следует, что при добавлении к обеим последовательностям слева по одному символу число общих R-вложений увеличивается вдвое, если добавляются равные символы, и не меняется в противном случае. Поскольку $|R(x) \cap \cap R(())| = |R(()) \cap R(y)| = |\varepsilon| = 1$, а $|\mu\delta(x, y)|$ равно числу совпадающих символов в одинаковых позициях x и y при отсчете слева направо, имеем $R_0(x, y) = |R(x) \cap R(y)| = 2^{|\mu\delta(x, y)|}$, что и требовалось доказать.

□

Теорема 11 определяет алгоритм вычисления $R_0(x, y)$. Сложность алгоритма равна сложности вычисления $\mu\delta(x, y)$, которая, очевидно, для $m \le n$ равна $\mathbf{O}(m)$, плюс сложность возведения числа два в степень $|\mu\delta(x, y)|$, которая тоже равна $\mathbf{O}(m)$. Тем самым сложность алгоритма равна $\mathbf{O}(m)$.

6.2. Сумма μ-длин общих R-вложений

Теорема 12. $R_1(x, y) = R_1(x[2..m], y[2..n]))$, если $x_1 \ne y_1$,

$R_1(x, y) = 2R_1(x[2..m], y[2..n] + R_0(x[2..m], y[2..n])$, если $x_1 = y_1$.

Доказательство.

Обозначим $R = R(x) \cap R(y)$ и $R_{-1} = R(x[2..m]) \cap \cap R(y[2..n])$. Если $x_1 \ne y_1$, то R-вложению $u \in R_{-1}$ соответствует одно R-вложение $\varepsilon u \in R$ той же μ-длины. Тем самым, $R_1(x, y) = \Sigma\{|\mu(u)| : u \in R\} = = \Sigma\{|\mu(u)| : u \in R_{-1}\} = R_1(x[2..m], y[2..n]))$. Если $x_1 = y_1$, то R-вложению $u \in R_{-1}$ соответствуют два R-вложения $\varepsilon u \in R$ той же μ-длины и $x_1 u \in R$ с μ-длиной на 1 большей. Тем самым, $R_1(x, y) = \Sigma\{|\mu(u)| : u \in \in R\} = \Sigma\{|\mu(u)| : u \in R_{-1}\} + (\Sigma\{|\mu(u)| : u \in R_{-1}\} + + R_0(x[2..m], y[2..n])) = 2\Sigma\{|\mu(u)| : u \in R_{-1}\} + + R_0(x[2..m], y[2..n]) = 2R_1(x[2..m], y[2..n] + + R_0(x[2..m], y[2..n])$.

□

Теорема 12 определяет алгоритм вычисления $R_1(x, y)$, сложность которого по порядку, очевидно, не превышает сложности алгоритма вычисления $R_0(x, y)$, т.е. для $m \le n$ равна $\mathbf{O}(m)$.

Теорема 13. $R_1(x, y) = 1*C_r^1 + 2*C_r^2 + ... + r*C_r^r$ (A001787), где $r = |\mu\delta(x, y)|$.

Доказательство.

Утверждение теоремы непосредственно следует из того факта, что число общих R-вложений μ-длины i равно $C^i_{|\mu\delta(x,y)|}$.

□

Теорема 13 определяет алгоритм вычисления $R_1(x, y)$. Сложность алгоритма равна сложности вычисления $\mu\delta(x, y)$, которая, очевидно, для $m \le n$ равна $\mathbf{O}(m)$, плюс сложность вычисления факториалов $i!$ для $i \in 0..r$, равная $\mathbf{O}(m)$, плюс сложность суммирования r чисел, равная $\mathbf{O}(m)$. Тем самым сложность алгоритма равна $\mathbf{O}(m)$.

6.3. Сумма минимумов и сумма произведений чисел E-вложений общих R-вложений

Теорема 14. $R_2(x, y) = R_3(x, y) = R_0(x, y)$.

Доказательство.

Утверждение теоремы непосредственно следует из того факта, что R-вложению $u \in R(x)$ соответствует ровно одно E-вложение v такое, что $\rho(v) = u$: $R_2(x, y) = \Sigma\{min\{|r(u, x)|, |r(u, y)|\} : u \in R(x) \cap \cap R(y)\} = \Sigma\{min\{1, 1\} : u \in R(x) \cap R(y)\} = |R(x) \cap \cap R(y)| = R_0(x, y)$; $R_3(x, y) = |R(x, y)| = |\cup \{r(u, x) \times \times r(u, y) : u \in R(x) \cap R(y)\}| = |R(x) \cap R(y)| = R_0(x, y)$.

□

Теорема 14 определяет алгоритм вычисления $R_2(x, y)$ и $R_3(x, y)$ той же сложности, что алгоритм вычисления $R_0(x, y)$, т.е. для $m \le n$ сложности $\mathbf{O}(m)$.

6.4. Функция похожести на основе наибольшей длины общего R-вложения

Теорема 15.

$lcR(x, y) = lcR(x[2..m], y[2..n]) + 1$, если $x_1 = y_1$;

$lcR(x, y) = lcR(x[2..m], y[2..n])$, если $x_1 \ne y_1$.

Доказательство. Функция $\delta(x, y)$ устанавливает позиционное соответствие совпадающих символов x и y при счете позиций слева направо. Поэтому $lcR(x, y) = |\mu\delta(x, y)|$. Отсюда непосредственно следует утверждение теоремы.

□

Теорема 15 определяет алгоритм вычисления функции $R_4(x, y) = lcR(x, y)$ сложности $\mathbf{O}(m)$ для $m \le n$.

7. *O*-ВЛОЖЕНИЯ

7.1. Число общих O-вложений

Для случая $x_m = y_n = h$ обозначим через $I = \{i \in \in 1..m : x_i = h\}$ и $J = \{j \in 1..n : y_j = h\}$ множества индексов по которым находится символ h в x и y, соответственно.

Обозначим для $i \in I, j \in J$: $L_{i,j} = L(x[i - 1]) \cap \cap L(y[j - 1])$.

Обозначим $K = (I \times J)\backslash\{(m, n)\}$ и $L_h(x, y) = = L_{m,n}\backslash\cup \{L_{i,j} : (i, j) \in K\}$.

Для $i \in I$ и $j \in J$ обозначим через $u_{i,j}$ максимальное общее L-вложение в $x[i - 1]$ и $y[j - 1]$: $u_{i,j} = = \lambda(v_{i,j})$, где $v_{i,j}$ общее E-вложение в $x[i - 1]$ и $y[j - 1]$, определяемое условием: $\forall t = 1..min\{i, j\} - 1$ ($x_{i-t} =$

$= y_{i-t} \Rightarrow v_{i,j}(i-t) = x_{i-t}$) & ($x_{i-t} \neq y_{i-t} \Rightarrow v_{i,j}(i-t) = \varepsilon$). Множество всех общих L-вложений в $x[i-1]$ и $y[j-1]$ равно $\lambda E(u_{i,j})$, т.е. получается из $u_{i,j}$ всеми возможными заменами некоторых символов на пустой символ и последующим удалением префикса пустых символов.

Лемма 1. Пусть $x_m = y_n = h$. Вычисление $|L_h(x, y)|$ эквивалентно вычислению числа слагаемых СДНФ некоторой конъюнкции дизъюнкций переменных без отрицаний, где число переменных равно числу непустых символов в максимальном общем L-вложении $u_{m,n}$ в $x[m-1]$ и $y[n-1]$.

Доказательство.

$L_h(x, y) = L_{m,n} \setminus \cup \{L_{i,j} : (i, j) \in K\} = L_{m,n} \setminus \cup \{L_{m,n} \cap L_{i,j} : (i,j) \in K\}$. Для получения $L_h(x, y)$ нам нужно из $L_{m,n}$ удалить множество $L_{m,n} \cap L_{i,j}$ для каждых $(i, j) \in K$.

Определим "пересечение" $u^{\wedge}_{i,j}$ вложений $u_{m,n}$ и $u_{i,j}$, имеющее длину $|u_{m,n}|$ и совпадающее с $u_{m,n}$ и $u_{i,j}$ в тех позициях, считая справа налево, в которых они совпадают между собой, а во всех остальных позициях содержит пустой символ: $|u^{\wedge}_{i,j}| = |u_{m,n}|$ и $\forall\, t = 1..|u_{m,n}|$ ($t > |u_{i,j}| \Rightarrow u^{\wedge}_{i,j}(|u_{m,n}| - t) = \varepsilon$) & ($t \le |u_{i,j}|$ & $u_{m,n}(|u_{m,n}| - t) = u_{i,j}(|u_{i,j}| - t) \Rightarrow u^{\wedge}_{i,j}(|u_{m,n}| - t) = u_{m,n}(|u_{m,n}| - t)$) & ($t \le |u_{i,j}|$ & $u_{m,n}(|u_{m,n}| - t) \neq u_{i,j}(|u_{i,j}| - t) \Rightarrow u^{\wedge}_{i,j}(|u_{m,n}| - t) = \varepsilon$). Очевидно, $u^{\wedge}_{m,n} = u_{m,n}$.

Удалим из вложения $u^{\wedge}_{i,j}$ символы в тех позициях, в которых $u_{m,n}$ содержит пустой символ, и обозначим результат $w_{i,j}$: если $u^{\wedge}_{i,j} = u_1 h_1 u_2 h_2 \ldots u_k h_{k-1} u_k$, $u_{m,n} = v_1 \varepsilon v_2 \varepsilon \ldots v_{k-1} \varepsilon v_k$ и для $i \in 1..k$ $|u_i| = |v_i|$ и $v_i \in H^*$, то $w_{i,j} = u_1 u_2 \ldots u_{k-1} u_k$. Очевидно, $w_{m,n} = \mu(u_{m,n})$.

Все $w_{i,j}$ имеют одинаковую длину $|w_{m,n}|$, равную числу непустых символов в $u_{m,n}$. Каждому L-вложению из $L_{m,n} \cap L_{i,j}$ взаимно-однозначно соответствует E-вложение в $w_{i,j}$, число таких вложений равно 2^k, где $k = |\mu(w_{i,j})|$ число непустых символов в $w_{i,j}$.

Поставим каждому $t \in 1..|w_{m,n}|$ в соответствие булеву переменную α_t, означающую, что в позиции t может быть непустой символ. Тогда E-вложения в $w_{i,j}$ задаются булевой функцией $F_{i,j} = \&\{\neg\alpha_t : t \in 1..|w_{m,n}| \;\&\; w_{i,j}(t) = \varepsilon\}$, которая принимает значение **true** на тех и только тех наборах α_1, ..., в которых $\alpha_t =$ **false** для всех тех индексов t, по которым в $w_{i,j}$ и, следовательно, в любом E-вложении в $w_{i,j}$ находится пустой символ. Если по индексу t в $w_{i,j}$ находится непустой символ, то в одних E-вложениях в $w_{i,j}$ в этой позиции будет пустой символ, а в других непустой символ, что означает, что функция $F_{i,j}$ не зависит от α_t. Очевидно, что $F_{m,n} = \&\varnothing =$ **true**.

Разность $L_{m,n} \setminus \cup \{L_{m,n} \cap L_{i,j} : (i, j) \in K\}$ задается булевой функцией $F = F_{m,n} \setminus \vee \{F_{i,j} : (i, j) \in K\} = \& \{\neg F_{i,j} : (i,j) \in K\}$, и $\neg F_{i,j} = \vee\{\alpha_t : t \in 1..|w_{m,n}| \;\&\; w_{i,j}(t) = \varepsilon\}$. Таким образом, F – это конъюнкция дизъюнкций переменных без отрицаний, и число $|L_h(x, y)|$ равно числу слагаемых в СДНФ функции F.

□

Теорема 16.

$O_0(x, y) = O_0(x[m-1], y) + O_0(x, y[n-1]) - O_0(x[m-1], y[n-1])$, если $x_m \neq y_n$.

$O_0(x, y) = O_0(x[m-1], y) + O_0(x, y[n-1]) - O_0(x[m-1], y[n-1]) + L_h(x, y])$, если $x_m = y_n$.

Доказательство.

Обозначим следующие множества пар E-вложений:

$E = O_r(x, y)$,

$E_{00} = O_r(x[m-1], y[n-1])(\varepsilon, \varepsilon)$ – пара последних элементов $(\varepsilon, \varepsilon)$,

$E_{01} = (O_r(x[m-1], y])(\varepsilon, ()))\setminus E_{00}$ – пара последних элементов (ε, y_n),

$E_{10} = (O_r(x, y[n-1])((), \varepsilon))\setminus E_{00}$ – пара последних элементов (x_m, ε),

$E_{11} = E\setminus(E_{00} \cup E_{01} \cup E_{10})$ – пара последних элементов (x_m, y_n), поскольку $E_{00} \cup E_{01} \cup E_{10}$ содержат все пары E-вложений общих O-вложений, в которых пара последних элементов содержит ε.

Очевидно, $E = E_{00} \cup E_{01} \cup E_{10} \cup E_{11}$. Пары последних элементов пар E-вложений из разных множеств E_{00}, E_{01}, E_{10}, E_{11} разные, поэтому эти множества попарно не пересекаются. Поэтому $|E| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{11}|$, $|E_{00} \cup E_{01}| = |E_{00}| + |E_{01}|$, $|E_{00} \cup E_{10}| = |E_{00}| + |E_{10}|$.

Поскольку для любых x, y имеет место $O_0(x, y) = |O(x) \cap O(y)| = |O_r(x, y)|$ и $L_0(x, y) = L_3(x, y) = |L(x, y)|$, утверждение теоремы можно переписать в виде:

$|E| = (|E_{00}| + |E_{01}|) + (|E_{00}| + |E_{10}|) - |E_{00}| = |E_{00}| + |E_{01}| + |E_{10}|$, если $x_m \neq y_n$,

$|E| = (|E_{00}| + |E_{01}|) + (|E_{00}| + |E_{10}|) - |E_{00}| + |L_h(x, y)| = |E_{00}| + |E_{01}| + |E_{10}| + |L_h(x, y)|$, если $x_m = y_n$.

Рассмотрим случай $x_m \neq y_n$. В этом случае $E_{11} = \varnothing$, что влечет $|E_{11}| = 0$ и $|E_{00}| = |E_{00}| + |E_{01}| + |E_{10}|$, что и требовалось доказать в этом случае.

Рассмотрим случай $x_m = y_n = h$. В этом случае каждая пара E-вложений из E_{11} имеет вид $(\varepsilon^{m-|v|-k-1} v \varepsilon^k h, \varepsilon^{n-|v|-k-1} v \varepsilon^k h)$, где $(\varepsilon^{m-|v|-k-1} v \varepsilon^k, \varepsilon^{n-|v|-k-1} v \varepsilon^k) \in L_h(x, y)$, т.е. при переходе от $x[m-1]$ и $y[n-1]$ к x и y эта пара образована добавлением справа символа h к паре E-вложений общих L-вложений в $x[m-1]$ и $y[n-1]$, но таких, которых не было раньше, т.е. которые не являются парой E-вложений общих L-вложений в $x[i-1]$ и $y[j-1]$, где $x_i = y_j = h$ и $i < m$ или $j < n$. Будем говорить, что пара $(\varepsilon^{m-|v|-k-1} v \varepsilon^k h,$

$\varepsilon^{n-|v|-k-1}v\varepsilon^k h$) соответствует паре ($\varepsilon^{m-|v|-k-1}v\varepsilon^k$, $\varepsilon^{n-|v|-k-1}v\varepsilon^k$). Это соответствие, очевидно, является биекцией множеств E_{11} и $L_h(x, y)$. Тем самым, $|E_{11}| = |L_h(x, y)|$. Поэтому $|E| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{11}| = |E_{00}| + |E_{01}| + |E_{10}| + |L_h(x, y)|$, что и требовалось доказать в этом случае.

□

Теорема 16 определяет алгоритм вычисления $O_0(x, y)$. Число шагов алгоритма равно $\mathbf{O}(mn)$, что определяется числом функций вида $O_0(x[m - i], y[n - j])$, где $i \in 0..m$ и $j \in 0..n$, при условии, что каждая функция вычисляется не более одного раза (после чего ее значение сохраняется). На каждом шаге вычисления имеют сложность $\mathbf{O}(1)$. Тем самым сложность алгоритма равна $\mathbf{O}(mn)$*$\mathbf{O}(L_h(x, y))$, где $\mathbf{O}(L_h(x, y))$ сложность вычисления функции $|L_h(x, y)|$.

Лемма 2. Пусть имеется k не обязательно различных множеств $a(s)$, $s \in 1..k$. Обозначим для $S \subseteq 1..k$, $S \neq \varnothing$, пересечение множеств $a(s)$, индексы которых пробегают множество S, через $c(S) = \cap\{a(s) : s \in S\}$ и сумму числа подмножеств множества $c(S)$ по всем S, для которых $|S| = l$, через $E(l) = \Sigma\{2^{|c(S)|} : S \subseteq 1..k \ \& \ |S| = l\}$. Тогда число различных множеств, вложенных хотя бы в одно из множеств $a(s)$, $s \in 1..k$, равно чередующейся сумме $E(1) - E(2) + E(3) - E(4) \ldots (-1)^{k+1}E(k)$, а сумма F размеров этих множеств, увеличенных на 1, равна чередующейся сумме $F(1) - F(2) + F(3) - F(4) \ldots (-1)^{k+1}F(k)$, где $F(l) = \Sigma\{(|c(S)| + 2)\, 2^{|c(S)|-1} : S \subseteq 1..k \ \& \ |S| = l\}$. Эти чередующиеся суммы могут быть вычислены за время $\mathbf{O}(2^k)$ при условии, что за время $\mathbf{O}(1)$ могут выполняться операции пересечения двух множеств (а также арифметические операции над целыми числами).

Доказательство.

Обозначим число различных множеств, вложенных хотя бы в одно из множеств $a(s)$, $s \in 1..k$, через $E(a(1), \ldots, a(k))$, а сумму их размеров, увеличенных на 1, через $F(a(1), \ldots, a(k))$.

Будем вести доказательство индукцией по k. Для $k = 1$ имеется единственное множество $S \subseteq 1..1$, $S \neq \varnothing$, а именно $S = \{1\}$, и $c(\{1\}) = \cap\{a(s) : s \in \{1\}\} = a(1)$. Число различных подмножеств множества $a(1)$ равно $2^{|a(1)|}$ и $E(a(1)) = E(1) = 2^{|a(1)|}$, а сумма их длин, увеличенных на 1, равна $1*C_r^0 + 2*C_r^1 + \ldots + r*C_r^{r-1} + (r+1)*C_r^r = (r+2)2^{r-1}$ (A001792), и $F(a(1)) = F(1) = (r+2)2^{r-1}$, где $r = |a(1)|$.

Пусть утверждение верно для k и докажем его для $k + 1$.

Число различных множеств, вложенных хотя бы в одно из множеств $a(s)$, где $s \in 1..k + 1$, равно числу различных множеств, вложенных хотя бы в одно из множеств $a(s)$, где $s \in 1..k$, т.е. $E(a(1), \ldots, a(k))$, плюс число различных множеств, вложенных в $a(k + 1)$, т.е. $2^{|a(k+1)|}$, кроме тех, что вложены в пересечение $a(k + 1)$ с объединением множеств $a(1), \ldots, a(k)$, число которых равно $E(a(k + 1) \cap \cap a(1), \ldots, a(k + 1) \cap a(k))$.

Обозначим $E_{k+1} = E(a(1), \ldots, a(k + 1))$, $E_k = E(a(1), \ldots, a(k))$, $E_k^\wedge = E(a(k+1) \cap a(1), \ldots, a(k+1) \cap \cap a(k))$. Имеем $E_{k+1} = E_k + 2^{|a(k+1)|} - E_k^\wedge$.

Рассмотрим $S \subseteq 1..k + 1$. Пусть $|S| = 1$, тогда $S = \{i\}$, $i \in 1..k + 1$. Для $i \in 1..k$ слагаемое $2^{|cS)|} = 2^{|a(i)|}$ один раз входит в сумму E_k со знаком "+", в результате для $i \in 1..k + 1$ слагаемое $2^{|cS)|} = 2^{|a(i)|}$ один раз входит в сумму E_{k+1} с тем же знаком. Пусть $|S| > 1$. Если $k + 1 \notin S$, то слагаемое $2^{|c(S)|}$ один раз входит в сумму E_k со знаком "$(-1)^{|S|+1}$", в результате для $i \in \in 1..k + 1$ слагаемое $2^{|c(S)|}$ один раз входит в сумму E_{k+1} с тем же знаком. Если $k + 1 \in S$, то $c(S) = \{a(i_1) \cap \ldots \cap a(i_{|S|-1}) \cap a(k + 1)\} = \{(a(k + 1) \cap \cap a(i_1)) \cap \ldots \cap (a(k + 1) \cap a(i_{|S|-1}))\}$. Поэтому слагаемое $2^{|c(S)|}$ один раз входит в сумму $E_k^\wedge$ со знаком "$(-1)^{|S|}$", но, поскольку сумма $E_k^\wedge$ вычитается, слагаемое $2^{|c(S)|}$ один раз входит в сумму E_{k+1} со знаком "$(-1)^{|S|+1}$".

Аналогично сумма длин, увеличенных на 1, различных множеств, вложенных хотя бы в одно из множеств $a(s)$, где $s \in 1..k + 1$, равно сумме длин, увеличенных на 1, различных множеств, вложенных хотя бы в одно из множеств $a(s)$, где $s \in 1..k$, т.е. $F(a(1), \ldots, a(k))$, плюс сумма длин, увеличенных на 1, различных множеств, вложенных в $a(k + 1)$, т.е. $(|a(k + 1)| + 2)2^{|a(k+1)|-1}$, кроме тех, что вложены в пересечение $a(k + 1)$ с объединением множеств $a(1), \ldots, a(k)$, сумма длин которых, увеличенных на 1, равна $F(a(k + 1) \cap a(1), \ldots, a(k + 1) \cap a(k))$. Обозначим $F_{k+1} = F(a(1), \ldots, a(k + 1))$, $F_k = F(a(1), \ldots, a(k))$, $F_k^\wedge = F(a(k + 1) \cap a(1), \ldots, a(k + 1) \cap a(k))$. Имеем $F_{k+1} = F_k + (|a(k + 1)| + 2)2^{|a(k+1)|-1} - F_k^\wedge$. Рассмотрим $S \subseteq 1..k + 1$. Пусть $|S| = 1$, тогда $S = \{i\}$, $i \in 1..k + 1$. Для $i \in 1..k$ слагаемое $(|c(S)| + 2)2^{|c(S)|-1} = (|a(i)| + 2)2^{|a(i)|-1}$ один раз входит в сумму F_k со знаком "+", в результате для $i \in 1..k + 1$ слагаемое $(|c(S)| + 2)2^{|c(S)|-1} = (|a(i)| + 2)2^{|a(i)|-1}$ один раз входит в сумму F_{k+1} с тем же знаком. Пусть $|S| > 1$. Если $k + 1 \notin S$, то слагаемое $(|c(S)| + 2)2^{|c(S)|-1}$ один раз входит в сумму F_k со знаком "$(-1)^{|S|+1}$", в результате для $i \in 1..k + 1$ слагаемое $(|c(S)| + 2)2^{|c(S)|-1}$ один раз входит в сумму F_{k+1} с тем же знаком. Если $k + 1 \in S$, то слагаемое $(|c(S)| + 2)2^{|c(S)|-1}$ один раз входит в сумму $F_k^\wedge$ со знаком "$(-1)^{|S|}$", но, поскольку сумма $F_k^\wedge$ вычитается, слагаемое $(|c(S)| + 2)2^{|c(S)|-1}$ один раз входит в сумму F_{k+1} со знаком "$(-1)^{|S|+1}$".

Утверждение доказано.

Число (не обязательно различных) множеств $c(S)$ равно числу непустых подмножеств S множества $1..k$, которое равно $2^k - 1$, поэтому сложность вычисления E равна $\mathbf{O}(2^k)$ при условии, что за время $\mathbf{O}(1)$ могут выполняться операции пересечения двух множеств (а также арифметические операции над целыми числами).

□

Обозначим множество индексов последовательности x, по которым находится символ h: $K_x(h) = \{i \in 1..m : x(i) = h\}$. Для $i \in K_x(h)$ обозначим множество пар (символ в x, позиция символа в x относительно позиции i), кроме пары $(h, 0)$: $P_x(i) = \{(x(t), t - i) : t \in 1..m \,\&\, t \neq i\}$. Пусть символ h входит $|K_x(h)| > 0$ раз в x и $|K_y(h)| > 0$ раз в y. Обозначим $k = |K_x(h)|*|K_y(h)|$. Для $i \in K_x(h)$ и $j \in K_y(h)$ для краткости обозначим $P(i,j) = P_x(i) \cap P_y(j)$. Для $S \subseteq K_x(h) \times K_y(h)$, $S \neq \varnothing$ обозначим $c(S) = \cap\{P(i,j) : (i,j) \in S\}$.

Теорема 17. Число различных общих O-вложений в x и y, содержащих символ h, равно чередующейся сумме $E = E(1) - E(2) + E(3) - E(4) \ldots (-1)^{k+1}E(k)$, где слагаемое $E(l)$ это сумма числа всех подмножеств множества $c(S)$ по всем S размера l, $|S| = l$: $E(l) = \Sigma\{2^{|c(S)|} : S \subseteq 1..k \,\&\, |S| = l\}$. Сложность вычисления равна $\mathbf{O}(n2^k)$.

Доказательство.

Рассмотрим общее O-вложение u и пару его E-вложений $v(x) \in o(u, x)$ и $v(y) \in o(u, y)$ таких, что $v(x)_i = x_i = h$ и $v(y)_j = y_j = h$. Этому взаимно-однозначно соответствует подмножество множества пар $P(i,j)$. Нам нужно вычислить число различных множеств, вложенных хотя бы в одно из множеств $P(i,j)$, где $i \in K_x(h)$ и $j \in K_y(h)$. Число таких множеств пар $P(i,j)$ равно k. По лемме 2 оно равно $E = E(1) - E(2) + E(3) - E(4) \ldots (-1)^{k+1}E(k)$, где $E(l) = \Sigma\{2^{|c(S)|} : S \subseteq 1..k \,\&\, |S| = l\}$, $l = 1..k$, сумма числа всех подмножеств множества $c(S)$ по всем $S \subseteq K_x(h) \times K_y(h)$ размера l, т.е. $|S| = l$, а $c(S) = \cap\{P(i, j) : (i,j) \in S\}$.

Просматриваем последовательность x длиной m, вычисляем $K_x(h)$. Просматривая $K_x(h)$, вычисляем множества $P_x(i)$. Вычисление множества $P_x(i)$ требует просмотра последовательности x длиной m. Тем самым, все множества $P_x(i)$, $i \in K_x(h)$, вычисляются за $\mathbf{O}(m|K_x(h)|)$. Аналогично все множества $P_y(j)$, $j \in K_y(h)$, вычисляются за $\mathbf{O}(n|K_x(h)|)$. Можно считать, что множество $P_x(i) = \{(x(t), t - i) : t \in 1..m \,\&\, t \neq i\}$ линейно упорядочено по возрастанию относительного индекса $t - i$; размер этого множества равен $m - 1$. Аналогично для множества $P_y(j)$; размер этого множества равен $n - 1$. Тогда для построения пересечения $P(i,j) = P_x(i) \cap P_y(j)$ требуется просмотр этих множеств, т.е. время $\mathbf{O}(n + m)$, а все множества $P(i,j)$, $i \in K_x(h)$ и $j \in K_y(h)$, вычисляются за время $\mathbf{O}(k(n + m))$. Аналогично каждое пересечение множеств $P(i, j)$ строится за время $\mathbf{O}(n + m)$. Сложность вычисления суммы E по лемме 2 равна $\mathbf{O}(2^k)$, но при условии, что пересечение двух множеств строится за время $\mathbf{O}(1)$, поэтому в данном случае потребуется время $\mathbf{O}((n + m)2^k)$. Общая сложность равна $\mathbf{O}(m) + \mathbf{O}(n) + \mathbf{O}(m|K_x(h)|) + \mathbf{O}(n|K_y(h)|) + \mathbf{O}(k(n + m)) + \mathbf{O}((n + m)2^k) = \mathbf{O}((n + m)2^k)$, что для $m \leq n$ равно $\mathbf{O}(n2^k)$.

□

Теорема 17 определяет следующий алгоритм вычисления $O_0(x, y)$:

1. SUMMA = 0.

2. Просматривая последовательность x, ищем непустой символ h, входящий в x.

2.1. Если не нашли, то конец алгоритма.

2.2. Если нашли, то ищем символ h в y.

2.2.1. Если не нашли, то в x заменяем h на пустой символ и переходим на п. 2.

2.2.2. Если нашли, то:

2.2.2.1. Вычисляем множество $K_x(h)$ индексов в x, по которым находится h, и множество $K_y(h)$ индексов в y, по которым находится h.

2.2.2.2. Для каждой пары $(i, j) \in K_x(h) \times K_y(h)$ строим множество пар $P(i,j)$.

2.2.2.3. Для каждого множества пар индексов $S \subseteq K_x(h) \times K_y(h)$, $S \neq \varnothing$ строим пересечение $c(S) = \cap\{P(i,j) : (i,j) \in S\}$.

2.2.2.4. Вычисляем $E = E(1) - E(2) + E(3) - E(4) \ldots (-1)^{k+1}E(k)$.

2.2.2.5. SUMMA = SUMMA + E.

2.2.2.6. В x и в y заменяем h на пустой символ и переходим на п. 2.

7.2. Сумма μ-длин общих O-вложений

Теорема 18. Сумма μ-длин различных общих O-вложений в x и y, содержащих символ h, равно чередующейся сумме $F = F(1) - F(2) + F(3) - F(4) \ldots (-1)^{k+1}F(k)$, где $F(l) = \Sigma\{(|c(S)| + 2)\, 2^{|c(S)|-1} : S \subseteq 1..k \,\&\, |S| = l\}$ сумма мощностей всех подмножеств всех множеств $c(S)$ для $|S| = l$. Сложность вычисления $\mathbf{O}(n2^k)$.

Доказательство аналогично доказательству теоремы 17.

□

Теорема 18 определяет следующий алгоритм вычисления $O_1(x, y)$:

1. Просматривая последовательность x, ищем непустой символ h, входящий в x.

1.1. Если не нашли, то конец алгоритма.

1.2. Если нашли, то ищем символ h в y.

1.2.1. Если не нашли, то в x заменяем h на пустой символ и переходим на п. 2.

1.2.2. Если нашли, то:

1.2.2.1. Вычисляем множество $K_x(h)$ индексов в x, по которым находится h, и множество $K_y(h)$ индексов в y, по которым находится h.

1.2.2.2. Для каждой пары $(i, j) \in K_x(h) \times K_y(h)$ строим множество пар $P(i, j)$.

1.2.2.3. Для каждого множества пар индексов $S \subseteq K_x(h) \times K_y(h)$, $S \neq \varnothing$ строим пересечение $c(S) = \cap \{P(i, j) : (i, j) \in S\}$.

1.2.2.4. Вычисляем $F = F(1) - F(2) + F(3) - F(4) \ldots (-1)^{k+1}F(k)$.

1.2.2.5. SUMMA = SUMMA + F.

1.2.2.6. В x и в y заменяем h на пустой символ и переходим на п. 2.

7.3. Сумма минимумов чисел E-вложений общих O-вложений

Теорема 19. Сумма минимумов чисел E-вложений общих O-вложений в x и y, содержащих символ h, равна $\Sigma\{min\{|I|, |J|\} * 2^{|A(I, J)|-1} : I \subseteq K_x(h)\ \&\ I \neq \varnothing\ \&\ J \subseteq K_y(h)\ \&\ J \neq \varnothing\}$, где $A(I, J) = (\cap\{P(i, j) : i \in I, j \in J\}) \setminus (\cup\{P(i, j) : i \in K_x(h) \setminus I \vee j \in K_y(h) \setminus J\})$.

Сложность вычисления равна $\mathbf{O}(n2^k)$.

Доказательство. Множество $A(I, J)$ представляет все общие O-вложения, содержащие символ h, которые имеют E-вложения в x, представляемые множеством I, и E-вложения в y, представляемые множеством J. Для каждого из таких O-вложений минимум чисел их E-вложений в x и в y равен $min\{|I|, |J|\}$. Число таких O-вложений равно $2^{|A(I, J)|}$, поэтому сумма минимумов чисел E-вложений таких общих O-вложений равна $min\{|I|, |J|\} * 2^{|A(I, J)|}$. Суммируя по всем парам множеств $I \subseteq K_x(h)$, $I \neq \varnothing$ и $J \subseteq K_y(h)$, $J \neq \varnothing$, получаем искомую сумму минимумов чисел E-вложений общих O-вложений в x и y, содержащих символ h.

При вычислении этой суммы операции сложения, умножения, возведения в степень числа 2 и вычисление минимума из двух чисел выполняются $\mathbf{O}(2^k)$ раз. Для вычисления всех $A(I, J)$ операции вычисления разности двух множеств, пересечения двух множеств, объединения двух множеств выполняются $\mathbf{O}(2^k)$ раз, а каждая такая операция выполняется за время $\mathbf{O}(n + m)$.

Просматриваем последовательность x длиной m, вычисляем $K_x(h)$. Просматривая $K_y(h)$, вычисляем множества $P_x(i)$. Вычисление множества $P_x(i)$ требует просмотра последовательности x длиной m. Тем самым, все множества $P_x(i)$, $i \in K_x(h)$, вычисляются за $\mathbf{O}(m|K_x(h)|)$. Аналогично все множества $P_y(j)$, $j \in K_y(h)$, вычисляются за $\mathbf{O}(n|K_y(h)|)$. Можно считать, что множество $P_x(i) = \{(x(t), t - i) : t \in 1..m\ \&\ t \neq i\}$ линейно упорядочено по возрастанию относительного индекса $t - i$; размер этого множества равен $m - 1$. Аналогично для множества $P_y(j)$; размер этого множества равен $n - 1$. Тогда для построения пересечения $P(i, j) = P_x(i) \cap P_y(j)$ требуется просмотр этих множеств, т.е. время $\mathbf{O}(n + m)$, а все множества $P(i, j)$, $i \in K_x(h)$ и $j \in K_y(h)$, вычисляются за время $\mathbf{O}(k(n + m))$. Для вычисления множества $A(I, J)$ операции вычисления разности двух множеств, пересечения двух множеств, объединения двух множеств выполняются за время $\mathbf{O}(n + m)$, а число таких множеств $A(I, J)$ равно $\mathbf{O}(2^k)$. В результате все множества $A(I, J)$ строятся за время $\mathbf{O}((n + m)2^k)$, после чего для вычисления суммы арифметические операции выполняются $\mathbf{O}(2^k)$ раз. Общая сложность равна $\mathbf{O}(k(n + m)) + \mathbf{O}((n + m)2^k) + \mathbf{O}(2^k) = \mathbf{O}((n + m)2^k)$, что для $m \leq n$ равно $\mathbf{O}(n2^k)$.

□

Теорема 19 определяет следующий алгоритм вычисления $O_2(x, y)$:

1. Просматривая последовательность x, ищем непустой символ h, входящий в x.

1.1. Если не нашли, то конец алгоритма.

1.2. Если нашли, то ищем символ h в y.

1.2.1. Если не нашли, то в x заменяем h на пустой символ и переходим на п. 2.

1.2.2. Если нашли, то:

1.2.2.1. Вычисляем множество $K_x(h)$ индексов в x, по которым находится h, и множество $K_y(h)$ индексов в y, по которым находится h.

1.2.2.2. Для каждой пары $(i, j) \in K_x(h) \times K_y(h)$ строим множество пар $P(i, j)$.

1.2.2.3. Для каждых $I \subseteq K_x(h)$, $I \neq \varnothing$, $J \subseteq K_y(h)$, $J \neq \varnothing$ строим $A(I, J)$.

1.2.2.4. Вычисляем $M = \Sigma\{min\{|I|, |J|\} * 2^{|A(I, J)|-1} : I \subseteq K_x(h)\ \&\ I \neq \varnothing\ \&\ J \subseteq K_y(h)\ \&\ J \neq \varnothing\}$.

1.2.2.5. SUMMA = SUMMA + M.

1.2.2.6. В x и в y заменяем h на пустой символ и переходим на п. 2.

7.4. Сумма произведений чисел E-вложений общих O-вложений

Теорема 20. $O_3(x, y) = O_3(x[m - 1], y) + O_3(x, y[n - 1]) - O_3(x[m - 1], y[n - 1])$, если $x_m \neq y_n$;

$O_3(x, y) = O_3(x[m - 1], y) + O_3(x, y[n - 1]) - O_3(x[m - 1], y[n - 1]) + L_3(x[m - 1], y[n - 1])$, если $x_m = y_n$.

Доказательство. Обозначим следующие множества пар E-вложений:

$E = O(x, y)$,

$L_{00} = L(x[m - 1], y[n - 1])$,

$E_{00} = O(x[m-1], y[n-1])(\varepsilon, \varepsilon)$ – пара последних элементов $(\varepsilon, \varepsilon)$,

$E_{01} = (O(x[m-1], y])(\varepsilon, ()))\backslash E_{00}$ – пара последних элементов (ε, y_n),

$E_{10} = (O(x, y[n-1])((), \varepsilon))\backslash E_{00}$ – пара последних элементов (x_m, ε),

$E_{11} = E\backslash(E_{00} \cup E_{01} \cup E_{10})$ – пара последних элементов (x_m, y_n), поскольку $E_{00} \cup E_{01} \cup E_{10}$ содержат все пары E-вложений общих O-вложений, в которых пара последних элементов содержит ε.

Очевидно, $E = E_{00} \cup E_{01} \cup E_{10} \cup E_{11}$. Пары последних элементов пар E-вложений из разных множеств E_{00}, E_{01}, E_{10}, E_{11} разные, поэтому эти множества попарно не пересекаются. Поэтому $|E| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{11}|$, $|E_{00} \cup E_{01}| = |E_{00}| + |E_{01}|$, $|E_{00} \cup E_{10}| = |E_{00}| + |E_{10}|$.

В этих обозначениях утверждение теоремы имеет вид

$|E| = (|E_{00}| + |E_{01}|) + (|E_{00}| + |E_{10}|) - |E_{00}| = |E_{00}| + |E_{01}| + |E_{10}|$, если $x_m \neq y_n$,

$|E| = (|E_{00}| + |E_{01}|) + (|E_{00}| + |E_{10}|) - |E_{00}| + |L_{00}| = |E_{00}| + |E_{01}| + |E_{10}| + |L_{00}|$, если $x_m = y_n$.

Рассмотрим случай $x_m \neq y_n$. В этом случае $E_{11} = \varnothing$, что влечет $|E_{11}| = 0$ и $|E_{00}| = |E_{00}| + |E_{01}| + |E_{10}|$, что и требовалось доказать в этом случае.

Рассмотрим случай $x_m = y_n = h$. Каждая пара E-вложений из E_{11} имеет вид $(\varepsilon^{m-|v|-k-1}v\varepsilon^k h, \varepsilon^{n-|v|-k-1}v\varepsilon^k h)$, где $(\varepsilon^{m-|v|-k-1}v\varepsilon^k, \varepsilon^{n-|v|-k-1}v\varepsilon^k) \in L_{00}$. Будем говорить, что пара $(\varepsilon^{m-|v|-k-1}v\varepsilon^k h, \varepsilon^{n-|v|-k-1}v\varepsilon^k h)$ соответствует паре $(\varepsilon^{m-|v|-k-1}v\varepsilon^k, \varepsilon^{n-|v|-k-1}v\varepsilon^k)$. Это соответствие, очевидно, является биекцией множеств E_{11} и L_{00}. Тем самым, $|E_{11}| = |L_{00}|$. Поэтому $|E| = |E_{00}| + |E_{01}| + |E_{10}| + |E_{11}| = |E_{00}| + |E_{01}| + |E_{10}| + |L_{00}|$, что и требовалось доказать в этом случае.

□

Теорема 20 определяет алгоритм вычисления $O_3(x, y)$. Число шагов алгоритма равно $\mathbf{O}(mn)$, что определяется числом функций вида $O_3(x[m-i], y[n-j])$ и $L_3(x[m-i], y[n-i])$, где $i \in 0..m$ и $j \in 0..n$, при условии, что каждая функция вычисляется не более одного раза (после чего ее значение сохраняется). На каждом шаге вычисления имеют сложность $\mathbf{O}(1)$. Тем самым сложность алгоритма равна $\mathbf{O}(mn)$.

7.5. Функция похожести на основе наибольшей длины общего O-вложения

Теорема 21.

$lcO(x, y) = max\{lcL(x[m-1], y[n-1]) + 1, lcO(x[m-1], y[n-1])\}$, если $x_m = y_n$:

$lcO(x, y) = max\{lcO(x, y[n-1]), lcO(x[m-1], y)\}$, если $x_m \neq y_n$.

Доказательство. Если $x_m = y_n$, то самые правые E-вложения общего O-вложения могут иметь в позиции m в x и в позиции n в y либо 1) символ $x_m = y_n$, либо 2) пустой символ. Если общее O-вложение наибольшей длины относится к случаю 1, то оно имеет вид ux_m, где u общее L-вложение префиксов $x[m-1]$ и $y[n-1]$, т.е. его μ-длина на 1 больше наибольшей длины общего L-вложения префиксов $x[m-1]$ и $y[n-1]$. Если общее O-вложение наибольшей длины относится к случаю 2, то оно является O-вложением наибольшей длины префиксов $x[m-1]$ и $y[n-1]$ и имеет такую же μ-длину. Отсюда следует утверждение теоремы для случая $x_m = y_n$.

Если $x_m \neq y_n$, то самые правые E-вложения общего O-вложения имеют пустой символ либо 1) в позиции m в x, либо 2) в позиции n в y. Если общее O-вложение наибольшей длины относится к случаю 1, то оно является O-вложением наибольшей длины префикса $x[m-1]$ и последовательности y и имеет такую же μ-длину. Если общее O-вложение наибольшей длины относится к случаю 2, то оно является O-вложением наибольшей длины последовательности x и префикса $y[n-1]$ и имеет такую же μ-длину. Отсюда следует утверждение теоремы для случая $x_m \neq y_n$.

□

Теорема 21 определяет алгоритм вычисления функции $O_4(x, y) = lcO(x, y)$ сложности $\mathbf{O}(mn)$.

8. ЗАКЛЮЧЕНИЕ

Некоторые из введенных нами функций подобия играют вспомогательную роль. Например, L_3 используется для вычисления O_3, а lcL используется для вычисления lcO, A_0 имеет как самостоятельное значение, так и используется для вычисления A_1. Хотя могут быть приложения, в которых эти вспомогательные функции окажутся как раз основными. L-вложения полезны там, где важно не только расстояние между символами вложения, но и расстояние от последнего символа вложения до конца последовательности, соответственно, R-вложения полезны там, где важно расстояние от начала последовательности до первого символа вложения.

Сравнивая разные типы функций (независимо от типа вложения), можно отметить следующее. Число общих вложений (функция 0) является хорошей числовой характеристикой, но она не учитывает длины вложений. Например, последовательности 11112222 и 1122111 имеют 9 общих подпоследовательностей (включая пустую), а сумма их длин равна 17, в то же время первая последовательность имеет с последовательностью 22221111

тоже 9 общих подпоследовательностей, но сумма их длин равна 20 за счет того, что есть две длинные подпоследовательности 1111 и 2222. Поэтому сумма длин общих вложений (функция 1) имеет самостоятельное значение.

Обе эти характеристики (функции 0 и 1) не учитывают тот факт, что одно общее вложение может входить в одну последовательность много раз, а в другую мало. Это пытается учесть сумма числа пар вхождений общих вложений (сумма произведений числа вхождений общих вложений – функция 3). Для того же примера: последовательности 11112222 и 1122111 имеют 279 пар вхождений общих вложений, а последовательности 11112222 и 22221111 – 139 пар за счет того, что в первой паре последовательностей очень много вхождений в обе последовательности общих вложений 12, 112 и 122, отсутствующих для второй пары.

В то же время функция 3 не удовлетворяет естественной *аксиоме направленности (direction) сходства* [5], иначе называемой *ограниченность самоподобием (bounded by self-similarity)* [4]: $f(x, y) \le min\{f(x, x), f(y, y)\}$. В частности эта функция строго возрастает, когда одна и та же непустая последовательность x сравнивается с последовательностями xx, xxx, $xxxx$, ... Этот недостаток преодолевает функция 4 – сумма минимумов числа вхождений общих вложений. К сожалению, эта функция плохо поддается алгоритмической оптимизации, в частности, для A-вложений, т.е. подпоследовательностей, мы не знаем алгоритма, отличного от полного перебора. Частичную оптимизацию можно было бы провести на основе эффективного алгоритма перечисления общих подпоследовательностей сложности $C(x, y)$ меньшей, чем сложность полного перебора. Поскольку вычисление числа вхождений данного вложения u в данную последовательность x имеет сложность $\mathbf{O}(|u|*|x|)$ [3, лемма 8], мы имели бы алгоритм вычисления функции 4 сложности $C(x, y)*\mathbf{O}(mn)$. Также если бы был эффективный (возможно, на подклассе последовательностей) алгоритм перечисления подпоследовательностей только одной данной последовательности сложности $C_1(x, y)$, то мы имели бы алгоритм вычисления функции 4 сложности $C_1(x, y)*\mathbf{O}(mn)$.

Функция 5, основанная на наибольшей длине общего вложения, также удовлетворяет аксиоме направленности. Но у нее тот недостаток, что не учитываются общие вложения, не являющиеся частью наибольшего (по длине) общего вложения. В нашем примере последовательности 11112222 и 1122111 имеют наибольшую общую подпоследовательность 1122, частью которой не является подпоследовательность 111, а последовательности 11112222 и 22221111 имеют две наибольшие общие подпоследовательности: 1111, не учитывающую все подпоследовательности из двоек, и 2222, не учитывающую все подпоследовательности из единиц.

Проблема перечисления общих вложений также исследуется. В качестве примера можно привести работу [6], в которой предлагается алгоритм перечисления максимальных по вложенности (а не по длине) общих подпоследовательностей, которому требуется полиномиальные время и память на каждую найденную максимальную общую подпоследовательность. Такие максимальные общие вложения обладают тем полезным свойством, что каждое общее вложение является частью одного или нескольких максимальных вложений. Можно было бы предложить функцию подобия, основанную на максимальных по вложенности общих вложениях: число таких вложений, сумма их длин и др. Это могло бы стать темой дальнейших исследований.

Другим направлением дальнейших исследований могли бы стать функции подобия последовательностей в алфавите, в котором символам приписаны те или иные веса. Пустому символу можно было бы приписать нулевой вес. Соответственно, вложению соответствует не число входящих в него символов, а сумма их весов. Отрицательный вес мог бы означать “штраф” за использование такого символа в общем вложении.

СПИСОК ЛИТЕРАТУРЫ

1. *Wagner R., Fischer M.* The string-to-string correction problem // Journal of the ACM. 1974. V. 21. № 1. P. 168–173. https://dl.acm.org/doi/10.1145/321796.321811
2. *Wang H.* All common subsequences, in: M.M. Veloso (Ed.), IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, 2007. https://www.aaai.org/Papers/IJCAI/2007/IJCAI07-101.pdf
3. *Cees Elzinga, Sven Rahmann, Hui Wang*: Algorithms for Subsequence Combinatorics. Theoretical Computer Science. 2008. V. 409. № 3. P. 394–404. https://doi.org/10.1016/j.tcs.2008.08.035
4. *Gilbert Ritschard and Matthias Studer* (*editors*). Proceedings of the International Conference on Sequence Analysis and Related Methods (LaCOSA II). Lausanne, Switzerland, June 8–10, 2016. https://www.academia.edu/83294569/Proceedings_of_the_International_Conference_on_Sequence_Analysis_and_Related_Methods_LaCOSA_II_Lausanne_Switzerland_-June_8_10_2016
5. *Знаменский С.В.* Модель и аксиомы метрик сходства, Программные системы: теория и приложения. 2017. Т. 8. Вып. 4. С. 347–357. https://doi.org/10.25209/2079-3316-2017-8-4-347-357
6. *Conte A., Grossi R., Punzi G. et al.* Enumeration of Maximal Common Subsequences Between Two Strings // Algorithmica. 2022. V. 84. P. 757–783. https://doi.org/10.1007/s00453-021-00898-5

TWENTY SIMILARITY FUNCTIONS OF TWO FINITE SEQUENCES

© 2023 г. I. Burdonov[a,#], and A. Maksimov[a,##]

[a] *Ivannikov Institute for System Programming of the Russian Academy of Sciences, Moscow, Russia*

[#] *e-mail: igor@ispras.ru*

[##] *e-mail: andrew@ispras.ru*

The article discusses the various numerical functions that determine the degree of "similarity" of the two given final sequences. These similarity measures are based on the concept we define of embedding in a sequence. A special case of such an attachment is the usual sub-subsequence. Other cases further require equality of distances between adjacent sub-sequence symbols in both sequences. This is generalization of the concept of a sequence segment (substring) in which these distances are unit. In addition, equality of distances from the beginning of the sequences to the first embedding symbol or from the last embedding symbol to the end of the sequences may be required. Except these last two cases, the attachment can be in a sequence several times. The literature uses functions such as the number of common attachments or the number of attachment occurrence pairs in a sequence. In addition to them, we enter three more functions: the sum of the lengths of total investments, the sum of the minima of the number of occurrences of a common embedding in both sequences and the similarity function based on the largest number of symbols of the common embedding. In total, 20 numerical functions are considered, for 17 of which algorithms (including new ones) of polynomial complexity are proposed, for two more functions, algorithms have exponential complexity with reduced a measure of degree. The Conclusion gives a brief comparative description of these investments and functions.

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 19–34

АНАЛИЗ ДАННЫХ

УДК 004.622

ПРИМЕНЕНИЕ ИМИТАЦИОННОГО КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ К ЗАДАЧЕ ОБЕЗЛИЧИВАНИЯ ПЕРСОНАЛЬНЫХ ДАННЫХ. МОДЕЛЬ И АЛГОРИТМ ОБЕЗЛИЧИВАНИЯ МЕТОДОМ СИНТЕЗА

© 2023 г. А. В. Борисов[a,*] (ORCID: 0000-0002-3124-2147), А. В. Босов[a,] (ORCID: 0000-0001-7163-341X), А. В. Иванов[a,***] (ORCID: 0000-0001-7811-7645)**

[a]*Федеральный исследовательский центр "Информатика и управление" РАН, 119333, Москва, ул. Вавилова, д. 44, кор. 2, Россия*

**e-mail: aborisov@ipiran.ru*

***e-mail: avbosov@ipiran.ru*

****e-mail: aivanov@ipiran.ru*

Поступила в редакцию 14.02.2023 г.
После доработки 12.03.2023 г.
Принята к публикации 14.05.2023 г.

Представлена вторая часть исследования, посвященного тематике автоматизированного обезличивания персональных данных. Обзор и анализ перспектив для исследований, выполненный ранее, здесь дополнен практическим результатом. Предложена модель процесса обезличивания, сводящая задачу обеспечения анонимности персональных данных к манипулированию выборками разнотипных случайных элементов. Соответственно, ключевой идеей преобразования данных для обеспечения их анонимности при условии сохранения полезности является применение метода синтеза, т.е. полной замены всех необезличенных данных синтетическими значениями. В предлагаемой модели выделен набор типов элементов, для которых предложены шаблоны синтеза. Совокупность шаблонов составляет алгоритм обезличивания методом синтеза. Методически каждый шаблон опирается на типовой статистический инструмент – частотные оценки вероятностей, ядерные оценки плотностей Розенблатта–Парзена, статистические средние и ковариации. Применение алгоритма иллюстрируется простым примером из области гражданских авиаперевозок.

DOI: 10.31857/S0132347423050023, **EDN:** ZXUVBM

1. ВВЕДЕНИЕ

Проблематика персональных данных в контексте их автоматизированной обработки в последние годы привлекала внимание исследователей, обладающих самыми разными компетенциями. Помимо очевидной содержательности для специалистов в областях права и информационной безопасности обезличивание персональных данных (ПД) и их последующая безопасная обработка стали источником постановок задач для специалистов в области анализа данных, распределенных вычислений, разработчиков баз данных, программистов и математиков. Причем разнообразие исследуемых вопросов оказалось очень широким, поэтому их содержательному обзору была посвящена отдельная работа [1]. Настоящая статья представляет вторую часть исследования, начатого в [1], и посвящена практическим результатам.

Анализ методов и алгоритмов обезличивания привел к двум принципиальным выводам. Во-первых, методов, обеспечивающих гарантированную анонимность обезличенным ПД, нет. Исключением может быть только метод синтеза, подразумевающий, что вместо исходного набора необезличенных ПД пользователю предоставляется набор синтетических данных заданного объема, которые связаны с исходным набором только совпадением некоторого набора характеристик, например, средних величин или размахов выборок.

Так, метод синтеза хорошо показал себя в задаче обезличивания строковых данных [2], особенности которых требуют для обезличивания больших усилий. Второй вывод – это реалистичность поддержки высокого уровня анонимности при обезличивании только числовых данных. Тексты (строки), потоковые, медийные и бинарные данные более-менее универсальных решений предложить не позволяют. Но несмотря на

очевидную привлекательность метода синтеза, в исследованиях ему уделяется довольно мало внимания. Вместе с тем даже самые простые статистические методы позволяют предлагать довольно универсальные решения для большинства возможных числовых атрибутов ПД. Такому алгоритму посвящена данная статья.

Ориентируясь на типовые статистические инструменты, семантику решаемой задаче и применяемым терминам дает, видимо самая распространенная концепция обезличивания *K*-анонимность [3, 4]. Но в отличие от многих известных реализаций этой концепции [5–13] предлагаемый подход ставит целью формирование результирующего обезличенного набора только из синтетических данных. Кроме того, сочетание статистики и *K*-анонимности дает возможность определить и вычислить все сопровождающие процесс обезличивания показатели, такие как уровни анонимности и полезности [14–18], а также используемые пользователями-аналитиками корреляции атрибутов ПД.

Статья организована следующим образом. В следующем разделе приведено описание модели процесса обезличивания – преобразования набора необезличенных ПД в набор обезличенных данных заданного объема. В третьем разделе приведен алгоритм обезличивания. Реализованный авторами прототип автоматизированной системы, модельные данные и результаты экспериментов обсуждаются в разделе 4. В заключение подведен итог исследования и кратко сформулированы возможные перспективы.

2. МОДЕЛЬ ОБЕЗЛИЧИВАНИЯ

2.1. Основные понятия

Предлагаемая для описания ПД модель использует несколько дополнительных понятий и терминов, которые объясняются в данном разделе. При моделировании процесса обезличивания следует исходить из того, что любые данные, которые отнесены законодательством к персональным, критически чувствительны к анонимности, поскольку позволяют идентифицировать субъекта ПД. Набор ПД может содержать множество различных свойств субъекта – данных определенного типа, называемых *атрибутами*. Предполагается, что эта совокупность атрибутов идентифицирует субъекта, так что их восстановление полностью или частично является деобезличиванием. При этом вклад разных атрибутов в идентификацию различен. Часть атрибутов может нести явный идентификационный характер, если атрибуты сами по себе являются прямыми указателями на конкретного субъекта ПД (типичный пример – паспортные данные). Такие атрибуты называются *персональными идентификаторами* и даже с самыми мягкими требованиями к анонимности при обезличивании должны исключаться. Аналогично следует поступать и с любыми другими чувствительными данными, которые не являются персональными идентификаторами, не могут быть использованы для прямой идентификации субъекта, но носят очевидный критический характер в связи с угрозой нарушения анонимности. Такие данные называют *чувствительными* (типичные примеры – пароли, коды, ключевые слова и т.п.) и они также должны исключаться при обезличивании.

После исключения из исходного необезличенного набора ПД чувствительной информации в нем остаются атрибуты, по отдельности или в совокупности представляющие угрозу нарушения анонимности. Предполагая, что степень угрозы от этих данных несопоставимо меньше, чем от персональных идентификаторов, все равно будем учитывать такую угрозу, называя такие атрибуты *квазиидентификаторами*. Можно считать, что квазиидентификаторы по отдельности являются гораздо менее чувствительными по отношению к анонимности, но влияние на анонимность больших совокупностей квазиидентификаторов преуменьшать нельзя [14, 16]. При этом именно атрибуты-квазиидентификаторы будут представлять значительный интерес для конечного потребителя обезличенных данных, т.е. будут подвергаться дальнейшей обработке, которая может привести к деобезличиванию. Соответственно, принципиально важными являются вопросы чувствительности квазиидентификаторов к анонимности, выбора применяемого алгоритма обезличивания и оценки защищенности от возможного нарушения анонимности обезличенных данных. Именно, на этих аспектах сосредоточено внимание в описанной далее модели и алгоритме обезличивания.

Определенный вызов представляет вопрос, а есть ли в необезличенном наборе данных те атрибуты, которые не являются квазиидентификаторами. Формальный ответ на этот вопрос – нет, потому что при наличии неограниченного ресурса для проведения идентификации субъекта ПД может быть использован любой атрибут. Да, знание малозначительного факта о субъекте не позволит его идентифицировать, но если этих фактов окажется очень много или к факту будут добавлены другие факты из дополнительных источников, то и малозначительный факт сыграет свою роль. Однако, чтобы эти рассуждения не привели к переоценке потенциальной угрозы, предлагается все-таки выделять в необезличенных ПД *нечувствительную* информацию, т.е. те данные, которые сколь-либо существенной угрозы анонимности не несут и могут быть использованы для идентификации субъекта лишь при наличии неограниченного ресурса.

В [1] упомянута небольшая часть резонансных фактов реализации угрозы нарушения анонимности обезличенных ПД. И эти факты, и многие другие не включали традиционного “бытового” понимания идентификации – точного установления личности. Исследователи справедливо считают, что возможность интерпретации выявленных ПД для установления личности доказана, а механизм этой интерпретации непринципиален. Например, определение атрибутов адреса и возраста субъекта, не означает его идентификации, нет фамильно-именной группы, тем более, номера паспорта или кредитной карты. Но субъекта по этим данным установить несложно, а нужные для этого методы и их законность в рамках научного исследования не обсуждаются. Важным для анализа защищенности от угрозы нарушения анонимности обезличенных данных является именно возможность выборки из множества обезличенных данных атрибутов одного или группы субъектов ПД. Таким образом, формулируя определение и формальное описание понятия деобезличивания, следует использовать именно такую интерпретацию идентификации. При этом для подлежащего обезличиванию обезличенного набора данных должна быть определена процедура оценки показателя защищенности ПД, который называется *уровнем анонимности*. Соответственно, следующий шаг – это применение к необезличенному набору такой процедуры обезличивания, чтобы уровень анонимности был велик настолько, насколько этого требует законодательство или/и владелец информации.

Понимая, что под *обезличиванием ПД понимаются действия, в результате которых становится невозможным без использования дополнительной информации определить принадлежность ПД конкретному субъекту ПД,* для предлагаемой математической модели требуется дать формальное определение “обратному” процессу деобезличивания, применимому для формирования объективной (числовой) оценки уровня анонимности обезличенных данных.

Пусть есть набор данных НД, содержащий необезличенные ПД вида

$$\text{НД} = \left\{\langle \text{ID-субъекта},\ \mathrm{A}_1,\ldots,\mathrm{A}_n\rangle_i\right\}_{i=1}^{N},$$

т.е. множество N однотипных кортежей, каждый из которых содержит данные по одному субъекту ПД, и эти данные представляют собой набор разнотипных атрибутов $\mathrm{A}_1,\ldots,\mathrm{A}_n$. Пусть далее в результате обезличивания сформирован набор данных

$$\text{ОД} = \left\{\langle \widehat{\text{ID}}\text{-субъекта},\ \hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_n\rangle_i\right\}_{i=1}^{N}.$$

Деобезличиванием данных ОД называется выборка по условиям, применяемым к атрибутам $\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_n$, из набора ОД любого подмножества $\left\{\langle \widehat{\text{ID}}\text{-субъекта}\rangle_j\right\}_{j=1}^{K}$ с целью точного или приближенного установления отвечающего этим идентификаторам набора атрибутов $\mathrm{A}_1,\ldots,\mathrm{A}_n$. Отметим, что в определении не фигурирует идентификатор ID-субъекта необезличенного набора, что отвечает сделанному замечанию о конкретном механизме интерпретации с целью идентификации субъекта ПД.

Следует отметить, что любая обработка обезличенных данных, сохраняющая идентификаторы субъектов, является деобезличиванием. Не приводит к деобезличиванию данных в смысле данного определения только такая обработка набора ОД, в результатах которой не сохраняются связи обработанных данных и исходных идентификаторов, т.е. обработка, включающая только операции агрегирования. Поскольку владелец НД и оператор, выполняющий обезличивание, не могут гарантированно контролировать содержание выполняемых над обезличенными данными операций, то формировать объективную оценку уровня защищенности обезличенных данных следует, исходя из потенциальной возможности любых выборок. Отсюда получаем следующее определение.

Путь при любой выборке, т.е. применении любых условий α к атрибутам данных ОД

$$\alpha = \left\{\hat{\mathrm{A}}_i \in \alpha_1,\ldots,\hat{\mathrm{A}}_n \in \alpha_n\right\},$$

где α_i – множество $\hat{\mathrm{A}}_i$ возможных значений $\hat{\mathrm{A}}_i$, из ОД формируется результирующий набор $\{\langle \widehat{\text{ID}}\text{-субъекта}\rangle_j\}_{j=1}^{K_\alpha}$ и существует число K такое, что $K \le K_\alpha$ для всех α. Такое число K называется *гарантированной оценкой уровня K-анонимности*.

Данное понятие следует концепции K-анонимности, но не означает, что предлагаемый далее алгоритм обезличивания направлен на обеспечение заданного показателя K-анонимности. Однако такая величина является хорошей объективной оценкой уровня анонимности. Величина K определяет минимальное число идентификаторов $\widehat{\text{ID}}$-субъекта, которое можно получить, максимально уточнив значения атрибутов $\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_n$. То есть какие бы действия ни выполнялись по уточнению атрибутов набора ОД, результирующий набор всегда будет содержать не менее K идентификаторов, так что, если целью обработки ОД является идентификация субъекта по признакам $\mathrm{A}_1,\ldots,\mathrm{A}_n$, то вероятность утери анонимности составит менее, чем $\frac{1}{K}$. И это верхняя гарантированная оценка, на реальную потерю анонимности влияет в сторону понижения и искажения

$\hat{A}_1,\ldots,\hat{A}_n$, и ограниченность в формировании условий $\alpha_1,\ldots,\alpha_n$, а главное – остающиеся "за скобками" действия по установлению связи между выявленными идентификаторами и верными атрибутами $A_1,\ldots,A_n$ и реальным субъектом ПД.

Данное понятие гарантированной оценки уровня анонимности K не зависит от размера N набора данных ОД. Так что в дополнение к нему имеет смысл рассматривать относительную характеристику уровня анонимности набора ОД. Именно, *относительным уровнем K-анонимности* называется величина $k = \frac{K}{N}100\%$. Получается, что 100%-обезличенные данные содержат все одинаковые атрибуты $\hat{A}_1,\ldots,\hat{A}_n$, что возможно только в том случае, если в ОД попадают только агрегаты по всему набору НД (это "идеальная" в смысле анонимности ситуация). При $k = \frac{100\%}{N}$ получается, что существует хотя бы один набор атрибутов $\hat{A}_1,\ldots,\hat{A}_n$, которому отвечает ровно один $\widehat{ID}$-субъекта (это наихудшая ситуация в смысле K-анонимности).

Величины K и k дают возможность формально определить следующие понятия. Набор данных ОД называется:

– *не допускающим деобезличивание*, если $k = 100\%$,

– *частично допускающим деобезличивание*, если $K > 1$,

– *допускающим идентификацию*, если $K = 1$.

Если при обезличивании формируется не допускающий деобезличивание набор ОД, то это означает фактическое отсутствие в ОД поля $\widehat{ID}$-субъекта, что возможно только в том случае, если ОД содержит только агрегированные данные (средняя температура, минимальная зарплата, максимальный тариф и т.п.). В таком случае выборки данных по заданным для деобезличивания требованиям достаточно для получения защищенного (с высоким уровнем анонимности) ОД и не требуются дополнительные алгоритмы обезличивания.

Любой набор ОД с полями $\widehat{ID}$-субъекта будет частично допускающим деобезличивание (иначе у всех записей должны совпадать значения всех атрибутов, что делает такой набор бесполезным). Если уровень анонимности, оцениваемый числами K и k для исходного набора данных НД, отобранного из имеющихся ПД по заданным требованиям к атрибутам, окажется неудовлетворительным, то требуется применение алгоритма обезличивания такого, чтобы уровень анонимности стал удовлетворительным. Если при этом результирующий ОД потеряет свойства применимости, то от предоставления ОД придется отказаться по причине невозможности обеспечить анонимность.

ОД, допускающий идентификацию – это такой набор данных, из которого можно выбрать ровно один $\widehat{ID}$-субъекта, задав некоторый набор условий $\alpha_1,\ldots,\alpha_n$. Такая выборка называется *идентификацией* и интерпретируется как реализация угрозы нарушения анонимности.

Величины K и k довольно грубо, но интуитивно понятно характеризуют набор данных с точки зрения угрозы нарушения анонимности. Грубость оценки как раз и обеспечивает ее гарантированный характер, т.е. ориентацию на наихудший случай из возможных. При этом нетрудно представить ситуацию, когда $K = 1$, но реальная угроза нарушения анонимности отсутствует. Эта ситуация может быть связана с любым атрибутом, имеющим очень большое множество возможных значений, вплоть до того, что все значения атрибута уникальны. Таким образом, указание любого из имеющихся значений $\langle\hat{A}_j\rangle$ такого атрибута $\hat{A}_j$ в качестве условия α_j будет идентификацией в смысле данного выше определения. Реальная же угроза нарушения анонимности может и не иметь места, т.к. на идентифицируемость знание $\langle\hat{A}_j\rangle$ может никак не влиять. Можно формализовать понятие такого невлияния. Для этого предлагается использовать следующее расширение понятия K-анонимности.

Если возможно для всех значений $\langle\hat{A}_j\rangle$ атрибута $\hat{A}_j$ задать ε-окрестность, т.е. интервал $\Delta_j = (\langle\hat{A}_j\rangle - \varepsilon_j, \langle\hat{A}_j\rangle + \varepsilon_j)$ с помощью малого числа ε_j такого, что семантические содержания значения $\langle\hat{A}_j\rangle$ и интервала Δ_j, т.е. равенств $\hat{A}_j = \langle\hat{A}_j\rangle$ и выражения $\hat{A}_j \in \Delta_j$, можно считать неотличимыми, то для определения оценки уровня анонимности согласно данному выше определению следует использовать условие

$$\alpha = \left\{\hat{A}_1 \in \alpha_1,\ldots,\hat{A}_j \in \Delta_j,\ldots,\hat{A}_n \in \alpha_n\right\},$$

для всех атрибутов $\hat{A}_j$, размер области значения которых существенно превосходит объем N имеющихся данных. Соответствующее число K_ε называется *гарантированной оценкой уровня ε-анонимности*. Основным претендентом на такую интерпретацию являются данные, область значений которых имеет непрерывную структуру. Сложность реализации этого понятия уровня анонимности будет состоять в том, что определение малости числа ε_j всегда будет носить некоторый субъективный характер.

Данные формальные определения трех оценок уровня анонимности (K, k, K_ε) потенциально да-

ют эксперту инструмент для качественной оценки уровня анонимности для конкретных наборов данных. Заметим, что нет формальных оснований для ограничения применимости этих характеристик только к обезличенным наборам, поэтому на практике оценки (K, k, K_ε) могут рассчитываться для любых наборов ПД как обезличенных, так и необезличенных (исходных).

При практическом применении предложенных характеристик в распоряжении эксперта окажется набор данных объемом N кортежей, перечень n атрибутов $A_1,\ldots,A_n$ и их типов. Для оценки уровня анонимности эксперт должен:

• исключить из атрибутов $A_1,\ldots,A_n$ персональные идентификаторы и чувствительную информацию;

• исключить из атрибутов $A_1,\ldots,A_n$ несущественные и нечувствительные атрибуты, не влияющие на оценку уровня анонимности;

• сформировать из оставшихся атрибутов перечень $A_1,\ldots,A_m$, отнеся к нему атрибуты, принимающие только числовые непрерывные значения;

• задав подходящую ε-окрестность для каждого атрибута, вычислить оценку K_ε (конкретные реализации расчета могут использовать разные варианты определения подходящего интервала Δ_j, например можно положить $\varepsilon_j = 0.05|\langle \hat{A}_j \rangle|$ или $\varepsilon_j = 0.05(x_{MAX} - x_{MIN})$, где x_{MAX}, x_{MIN} – максимальное и минимальное значение среди всех имеющихся значений $\langle \hat{A}_j \rangle$, что соответствует экспертной оценке 10% длины интервала Δ_j;

• сформировать из оставшихся атрибутов перечень $A_{m+1},\ldots,A_l$, отнеся к нему атрибуты, принимающие только числовые дискретные значения (сюда же следует относить и словарные типы) и вычислить оценку K;

• сравнить K_ε и K и, если обе величины принимают сопоставимые значения, вычислить оценку относительного уровня анонимности как $k = \dfrac{\min(K_\varepsilon, K)}{N} 100\%$.

2.2. Структура наборов необезличенных и обезличенных данных

Семантическое и структурное многообразие необезличенных ПД существенно затрудняют унификацию понятий и функций обработки данных, в т.ч. преобразований с целью обезличивания. Здесь представлен вариант унифицированного представления наборов данных НД и ОД, удобный для формального определения служебных структур и понятий, используемых далее в алгоритме обезличивания. Фактические представления необезличенных данных могут существенно отличаться, что предполагает наличие промежуточного функционального элемента, содержащего средства автоматизации, от которых потребуется выполнение преобразований:

$$\text{НД} \to \text{НД}(y) \to \text{ОД}(y) \to \text{ОД},$$

где НД – исходный набор ПД, ОД – целевой набор обезличенных ПД, НД(y), ОД(y) – исходный и целевой наборы данных в унифицированном представлении. Заметим, что через $\text{НД}(y) \to \text{ОД}(y)$ обозначено обезличивание данных, алгоритм которого является целью работы. Следуя введенной модели $\text{НД} = \{\langle \text{ID-персоны}, A_1,\ldots,A_n\rangle_i\}_{i=1}^N$, уточним ее. Будем предполагать, что структурное представление НД отвечает реляционной модели данных, т.е. обезличиваемые данные представлены в виде набора таблиц, таблицы обладают первичными ключами и используют вторичные ключи для установления связей, т.е. исходные данные реализованы типовой реляционной базой данных.

На рис. 1 показан простой, но довольно типичный пример структуры набора НД:

• таблица “Персональные идентификаторы” содержит все чувствительные атрибуты,

• набор “простых” атрибутов, представленных непосредственно своим значением, размещен в таблице “Атрибуты-значения” и связан ключом “Паспорт”,

• набор словарных атрибутов размещен в таблице “Атрибуты-словарные” и связан ключом “E-mail”,

• словарные значения определены своими ключами и данными в таблицах словарей “Словарь-3”, “Словарь-4”.

Элемент (строку, кортеж) НД(y), обозначенный выше $\langle$ID-субъекта, $A_1,\ldots,A_n\rangle$ составляют:

• ключ ID-субъекта, заменяющий всю совокупность атрибутов из “Персональные идентификаторы”,

• $A_1,\ldots,A_m$ – перечень “простых” атрибутов-значений,

• $A_{m+1},\ldots,A_n$ – перечень значений “словарных” атрибутов.

Если НД допускает атрибуты с множественными значениями, то такой НД корректно (полностью) может преобразовываться в НД(y), если

• m – это число атрибутов-значений, имеющихся в НД у субъекта, имеющего максимальное число атрибутов-значений, и допускается наличие в НД(y) пустых значений атрибутов $A_1,\ldots,A_m$,

• $(n-m)$ – это число словарных атрибутов, имеющихся в НД у субъекта, имеющего максимальное число словарных атрибутов, и допуска-

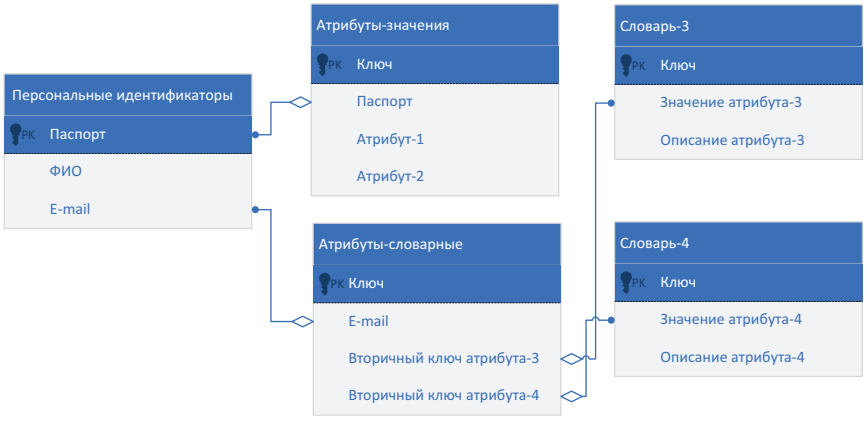

Рис. 1. Пример типового необезличенного набора.

ется наличие в НД(y) пустых значений атрибутов $A_{m+1},\ldots,A_n$.

Тогда набор данных вида $\{\langle$ID-субъекта, $A_1,\ldots,A_n\rangle_i\}_{i=1}^N$ называется *унифицированным набором необезличенных данных* НД(y).

Заметим, что такое определение является семантически зависимым, т.к. число атрибутов, определяющее структуру НД(y), зависит непосредственно от данных, содержащихся в НД. Это полезное свойство, т.к. дает возможность, во-первых, лучше понимать и контролировать потенциальные действия по деобезличиванию, во-вторых, удобнее оценивать уровень угрозы нарушения анонимности в ОД(y). Очевидным недостатком такого подхода к унификации НД являются потенциально значительные вычислительные трудности (требования к применяемым компьютерным ресурсам, необходимым при обезличивании больших объемов НД). НД(y), соответствующий примеру на рис. 1, может иметь вид

$$\left\{\begin{array}{c}\langle\text{ID-субъекта}\,(\text{GUID}),\ \text{Атрибут-1},\\ \text{Атрибут-2},\text{Значение атрибута-3},\\ \text{Значение атрибута-4}\rangle_i\end{array}\right\}_{i=1}^{N}.$$

Для ПД в унифицированном виде НД(y) уже начат процесс обезличивания, т.к. набор НД лишен персональных идентификаторов и дополнен уникальным идентификатором субъекта (все чувствительные атрибуты заменены не имеющим содержания ключом GUID – Globally Unique Identifier, гарантирующим при каждом новом формировании НД(y) новое значение, идентифицирующее запись НД). Причем для целей обезличивания подходящей будет любая из существующих реализаций GUID. Завершить первый этап обезличивания может служебная процедура, позволяющая в последующем выполнить деобезличивание. Для этого требуется набор данных

$$\text{Контракт} = \{\langle\text{ID-субъекта, ID-субъекта}\,(\text{GUID})\rangle_i\}_{i=1}^N.$$

Требуется или нет формировать набор Контракт, хранить его и иметь возможность деобезличивания в дальнейшем – зависит как от цели обезличивания, так и от статуса заказчика. Можно увидеть диаметрально разные ситуации, когда данные обезличиваются в интересах аналитики торговой сети (тогда ни о каком деобезличивании речи быть не может) и когда данные обезличиваются в интересах силового ведомства, выполняющего, к примеру, анализ нелегальных ресурсов в сети. Наконец, необходимо понимать, что сам факт наличия набора Контракт является источником угрозы нарушения анонимности.

Следующий шаг обезличивания НД также является формальным, но более содержательным. С НД(y) должны быть выполнены операции:

- при наличии заявленной цели обезличивания определены не отвечающие ей атрибуты ПД и соответствующие поля устранены как несущественные данные,
- определены квазиидентификационные данные.

Таким образом, в НД(y) останутся поля, которые определены в качестве квазиидентификаторов, и поля, которые признаны нечувствительными данными.

И наконец из структуры НД(y), к которой далее применяется алгоритм обезличивания, удаляются нечувствительные атрибуты. Если такие атрибуты есть, то они возвращаются неизменными уже в набор ОД(y).

2.3. Типы атрибутов

Исходя из того, что обезличивается набор данных вида $\{\langle$ID-субъекта, $A_1,\ldots,A_n\rangle_i\}_{i=1}^N$, содержа-

щий только квазиидентификаторы, остается определить только допустимые типы для атрибутов.

2.3.1. Числовое дискретное значение. Атрибут A_i набора НД(y) принимает числовое дискретное значение, если множество его потенциально возможных значений (т.е. не только в данном НД(y), но и в других возможных наборах) конечно или счетно, т.е. множество возможных значений можно перечислить и перенумеровать: $\langle A_i \rangle \in \{x_1, \ldots, x_l, \ldots\}$. Рассматривая величины x_l, надо понимать такие их свойства, как фактические значения, разницу между соседними величинами, равномерность или точки сгущения и т.п. Но кроме этих свойств принципиально важным является фактическое распределение значений $x_1, \ldots, x_l$, ... в данном конкретном наборе НД(y), т.е. то, как часто/редко те или иные значения x_l появляются в наборе НД(y).

2.3.2. Числовое непрерывное значение. Атрибут A_i набора НД(y) принимает числовое непрерывное значение, если множество его потенциально возможных значений (т.е. не только в данном НД(y), но и в других возможных наборах) задается диапазоном (интервалом) числовой оси: $\langle A_i \rangle \in \left(\left[x_{l_i}, x_{r_i}\right]\right)$. Здесь круглые скобки (,) или квадратные [,] используются при необходимости исключить или, наоборот включить, левую границу x_{l_i} или правую границу x_{r_i} в область возможных значений, при этом x_{l_i} может принимать значение $-\infty$, а x_{r_i} может принимать значение $+\infty$. Для дальнейших преобразований, выполняемых для обезличивания, принципиально важным является фактическое распределение значений такого атрибута A_i в данном конкретном наборе НД(y).

2.3.3. Словарное значение. По своему содержанию атрибут со словарным значением близок атрибуту с числовым дискретным значением, потому что для такого атрибута возможные значения можно пересчитать и пронумеровать. Более того, форма номера в словаре такого атрибута неважна, поэтому всегда можно считать, что значения такого атрибута $\langle A_i \rangle \in \{1, \ldots, n_i\}$, т.е. выбираются из натурального ряда, нумерующего словарь. Однако здесь есть важное отличие. Для числовых дискретных атрибутов принципиально важно конкретное числовое значение атрибута, т.е. всех величин x_l, для словарного это не имеет значения, поэтому они и нумеруются с помощью натурального ряда. Соответственно, выполнять обезличивание таких атрибутов придется способом, имеющим некоторые отличия от числовых дискретных атрибутов.

2.3.4. Дата/время. Темпоральные значения могут как нести самостоятельное содержание, например, дату рождения, время покупки, так и дополнять фактографию, представленную другими атрибутами, например, время изменения географического положения. Можно считать, что эти значения имеют вид день-месяц-год-час-минута-секунда, таким образом, тип дата/время тождественен вещественному числовому типу (типичное машинное представление для этого типа). Ясно, однако, что применять к нему те же методы, что для числового непрерывного значения неверно. Преобразования обезличивания здесь должны учитывать его интервальный характер (дней, месяцев, лет, часов, минут, секунд), а не распределение величин – значений соответствующего атрибута в конкретном НД(y).

2.3.5. Потоковые данные. Собственно значения этого типа представляют собой упорядоченные группы числовых величин (или структур, составленных из числовых величин), например, последовательности географических положений. Но особенностью этого типа является то, что его данные поступают постоянно, последовательно одно за другим в течение некоторого времени. Такого рода информация характерна для трекинговых систем, чатов, соцсетей и т.д. Обезличивание таких данных возможно в рамках предложенной модели, если допускается использовать вертикальное разбиение (обезличивание путем ограничения данных по каждому субъекту) или горизонтальное разбиение (обезличивание путем ограничения объема данных для субъектов).

2.3.6. Текстовые и бинарные данные. Текстовые данные – наиболее сложные с точки зрения обезличивания, унифицированных решений для них реализовать практически невозможно, любое решение будет исходить из конкретной семантики конкретного атрибута. Фиксированный формат строки решает этот вопрос, но, только в том случае, если текстовое значение атрибута гарантированно удовлетворяет некоторому формату. Примерами таких атрибутов могут быть адрес электронной почты, телефон, т.п. Также сюда можно отнести разного рода адресную информацию, например, адрес регистрации, проживания. Но даже в этих случаях обезличивание может выполняться только ограничением объема предоставляемой информации: для электронной почты – только домен, для телефона – только код и т.п. Если текст имеет произвольный формат, то он, по-видимому, не может быть гарантировано обезличен. Аналогична ситуация с бинарными данными. Если это квазиидентификаторы (а в модели НД(y) остались только такие атрибуты), то эти атрибуты должны исключаться.

Гарантированный способ обезличить как текстовые, так и бинарные данные – это замена их на синтетические. Успешные примеры такого рода хорошо известны – это примеры синтеза текста, синтеза изображений лиц и сцен. Сложность здесь состоит в том, что каждая задача такого рода

является существенным исследовательским вызовом и возможно даже представляет самостоятельный научный интерес. Соответственно, универсальной рекомендации по обезличиванию методом синтеза таких данных нет.

3. АЛГОРИТМ ОБЕЗЛИЧИВАНИЯ

3.1. Подготовка к обезличиванию

Описание основных преобразований предлагаемого алгоритма дается в предположении, что выполнены перечисленные в предыдущем разделе статьи подготовительные шаги, так что в обрабатываемом наборе НД(y) остались только атрибуты-квазидентификаторы перечисленных типов (2.3.1–2.3.5), в т.ч. при необходимости выполнена декомпозиция данных, и создан набор Контракт.

Сам алгоритм обезличивания состоит в последовательном применении *шаблонов обезличивания*, определенных далее для каждого типа атрибутов.

Упомянем здесь еще и финальный этап обезличивания – переход от унифицированного представления наборов данных, введенного для удобства описания алгоритма обезличивания, т.е. преобразование ОД(y) $\to$ ОД. Формальный смысл этого преобразования – вернуть исходную структуру набора, в частности, восстановить вторичные ключи, выделив и заполнив словари и другие сущности, характерные для исходной модели данных НД. Собственно, к обезличиванию этот переход прямого отношения не имеет, поэтому далее к нему возвращаться не будем.

3.2. Шаблоны обезличиввания

3.2.1. Числовое дискретное значение. Пусть из имеющегося набора атрибутов $A_1,\dots,A_n$ атрибуты $A_1,\dots,A_m$, т.е. для простоты первые m штук, принимают числовые дискретные значения, каждый из атрибутов в своей области допустимых значений. Будем считать, что каждый элемент набора НД(y), составленный из атрибутов $A_1,\dots,A_m$, является реализацией случайного вектора $\xi = \mathrm{col}(\xi_1,\dots,\xi_m)$, ξ_j – дискретная случайная величина со своей областью значений $\{x_j^1,\dots,x_j^{N(j)}\}$, $j = 1,\dots, m$. Выберем из НД(y) набор $\{\langle A_1,\dots,A_m\rangle_l\}_{l=1}^{L(m)}$ неповторяющихся реализаций ξ, упорядоченный последовательно по значениям $\langle A_1\rangle,\dots,\langle A_m\rangle$. Дополним полученный набор порядковым номером l, принимающим значения $0,\dots,L(m)-1$. Величина $l(m)$ зависит от набора данных НД(y) и от числа m участвующих атрибутов. Заметим, что эта величина не является произведением $N(j)$, $j = 1,\dots,m$, потому что не все комбинации возможных значений $x_1^{l_1},\dots,x_m^{l_m}, l_1 = 1,..,N(1),\dots, l_m = 1,\dots, N(m)$ могут встретиться в НД(y) Далее будем обозначать эту величину просто L. Полученный набор $\{\langle l, A_1,\dots,A_m\rangle\}_{l=0}^{L-1}$ можно интерпретировать как множество значений $\{x_l\}_{l=0}^{L-1}$ дискретной случайной величины ξ, дополненный номерами значений. Для расчета обезличенного значения вместо ξ применяется *шаблон синтеза числовых дискретных значений*. Он состоит в восстановлении (оценке) по имеющимся данным распределения исходного набора значений атрибутов $A_1,\dots,A_m$ в наборе НД(y) и формировании в ОД(y) полностью синтетических значений. Для их моделирования набор $\{\langle A_1,\dots,A_m\rangle_l\}_{l=0}^{L-1}$ из НД(y) дополняется сначала частотами $\{\langle p_l, A_1,\dots,A_m\rangle_l\}_{l=0}^{L-1}$, где p_l – частота появления значения x_l атрибутов $A_1,\dots,A_m$ в наборе НД(y), т.е. $p_l = \dfrac{Num(x_l)}{N}$, где $Num(x_l)$ – число появлений реализаций x_l в наборе НД(y). Эти величины могут быть вычислены вместе с формированием множества $\{A_1,\dots,A_{ml}\}_{l=0}^{L-1}$ неповторяющихся реализаций ξ. Записи для набора ОД(y) формируются путем моделирования псевдослучайных величин $\xi = \mathrm{col}(\hat{A}_1,\dots,\hat{A}_m)$ в соответствии с дискретным распределением $\{(p_l, x_l)\}_{l=0}^{L-1}$ для каждой записи $\{\langle \text{ID-субъекта}(\mathrm{GUID}), A_1,\dots,A_m\rangle_i\}_{i=1}^{N}$ в наборе НД(y).

3.2.2. Числовое непрерывное значение. Пусть из имеющегося набора атрибутов $A_1,\dots,A_n$ атрибуты $A_1,\dots,A_m$, т.е. для простоты первые m штук, принимают числовые непрерывные значения, каждый из атрибутов в своей области допустимых значений. Будем считать, что каждый элемент набора НД(y), составленный из атрибутов $A_1,\dots,A_m$, является реализацией случайного вектора $\xi = \mathrm{col}(\xi_1,\dots,\xi_m)$, ξ_j – непрерывная случайная величина со своей областью значений $\xi_j \in [x_{l_j}, x_{r_j}]$, где x_{l_j} – минимальное значение атрибута A_j в наборе НД(y), x_{r_j} – максимальное. Для каждой ξ_j, $j = 1,\dots,m$, из значений $\{\langle A_j\rangle_i\}_{i=1}^{N}$ набора НД(y) вычислим оценку среднего и дисперсии:

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N}\langle A_j\rangle_i, \qquad \sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\langle A_j\rangle_i\right)^2 - \mu_j^2.$$

Заметим, что в отличие от числовых дискретных значений, в данном случае не нужны операции по выявлению неповторяющихся реализаций ξ, не нужно упорядочивать и нумеровать значения. Для расчета обезличенного значения вместо ξ применяется *шаблон синтеза числового непрерывного*

значения. Он состоит в восстановлении (оценке) по имеющимся данным распределения исходного набора значений атрибутов $A_1,\ldots,A_m$ в наборе НД(y) и формировании в ОД(y) полностью синтетических значений. Для их моделирования распределение вектора ξ (набора атрибутов $A_1,\ldots,A_m$) аппроксимируется ядерной оценкой Розенблатта–Парзена [19, 20] с дополнением эмпирического правила Сильвермана [21]. Это значит, что плотность вероятности $f_\xi(x_1,\ldots,x_m)$ вектора ξ приближенно представляется в виде

$$f_\xi(x_1,\ldots,x_m) \approx \frac{1}{N\sqrt{\det(H^T H)}} \times$$

$$\times \sum_{i=1}^{N} \Phi(x_1,\ldots,x_m; \mathrm{col}(\langle A_1,\ldots,A_m\rangle_i), H^2),$$

$$H^2 = \mathrm{diag}(h_1^2,\ldots,h_m^2), \quad h_j = \sqrt[m+4]{\left(\frac{4}{m+2}\right)} \frac{1}{\sqrt[m+4]{N}} \sigma_j,$$

где $\Phi(x_1,\ldots,x_m; M, \Sigma)$ – m-мерная гауссовская плотность вероятности с вектором средних значений M и ковариационной матрицей Σ. Таким образом, исходное распределение вектора ξ (набора атрибутов $A_1,\ldots,A_m$) из НД(y) представляется гауссовской смесью из N слагаемых (ровно столько, сколько элементов в НД(y)), слагаемые имеют математические ожидания, задаваемые истинными значениями $\langle \hat{A}_1,\ldots,\hat{A}_m\rangle_i$ атрибутов, ковариации для каждого слагаемого – диагональные матрицы (т.е. элементы соответствующих m-мерных векторов независимы), а дисперсии задаются эмпирически обоснованным "сужением" оцененной по данным набора НД(y) дисперсии вектора ξ.

Для моделирования синтетических значений атрибутов $A_1,\ldots,A_m$ для набора ОД(y): в i-м элементе набора $\langle$ID-субъекта (GUID), $A_1,\ldots,A_m\rangle_i$ для j-го атрибута A_j моделируется случайная величина τ, имеющая равновероятное распределение на множестве числе $\{1,\ldots,N\}$, моделируется стандартная гауссовская величина ε, вычисляется значение $\langle \hat{A}_j\rangle_i = \langle A_j\rangle_\tau + h_j\,\varepsilon$. Отметим, что такая схема крайне проста в практической реализации.

3.2.3. Словарное значение. Характер данных этого типа аналогичен типу числовое дискретное значение. Действительно, не ограничивая общности, можно считать, что если атрибут A_1 словарный, то его значения можно описать реализациями ξ_1 – дискретной случайной величины с областью значений $\{1,\ldots,n\}$. Отличие же состоит в том, что для данных этого типа не измеряется "расстояние" между реализациями в традиционном арифметическом смысле. Разница между реализациями определяется семантикой словаря, так что унифицированного числового представления не существует. Поскольку есть характеристика атрибута A_1 в качестве квазиидентификатора, то ключевым является его влияние на оценку уровня анонимности, формируемого ОД(y), поэтому здесь применяется *шаблон устранения аномалий*. Его действие таково. Задается уровень T аномальных значений. Это величина в процентах, характеризующая приемлемое число повторений значений словарных атрибутов, не представляющее угрозы нарушения анонимности. Интуитивно приемлемое начальное значение $T=10\%$ означает, что для рассматриваемого словаря, содержащего n уникальных значений, приемлемым считается наличие не менее $\frac{10}{n}\%$ элементов для каждого из имеющихся n значений. Заметим, что шаблон применяется к НД(y), поэтому все n возможных словарных значений в обезличиваемом наборе есть, в отличие от НД, где словарь мог быть наполнен и неиспользуемыми значениями. Для выбора величины T можно использовать расчет уровня анонимности по данному атрибуту, описанный далее.

При нарушении условия для выбранного уровня T словарные значения, для которых условие нарушено, исключаются из словаря. Вместо всех исключенных словарных значений вводится одно нейтральное значение – "не определено", "не задано", "неизвестно". Область значений величины ξ_1 изменяется на $\{1,\ldots,n-m\}$, где m – число исключенных словарных значений. Далее атрибут A_1 включается в состав атрибутов типа числовое дискретное значение и вместе с ними подвергается обработке шаблоном синтеза числового дискретного значения.

3.2.4. Дата/время. Для атрибута A_1 этого типа применяется *шаблон разбиения*. Этот шаблон решает две задачи: во-первых, маскируются данные, потенциально угрожающие нарушением анонимности, во-вторых, тип дата/время заменяется простым словарным типом. Для применения шаблона разбиения к атрибуту A_1 выполняется следующее. Определяется диапазон дат – имеющихся в наборе НД(y) значений атрибута A_1. На основании значений диапазона выбирается одно из интервальных разбиений (выполняется семплирование диапазона). Сформировать коллекцию разбиений (семплов) – задача применяемого автоматизированного средства. Начальный набор, например, может включать:

- вариант для дат рождения – {⟨1 год, 1–3 года, 3–7 лет, 7–12 лет, 12–17 лет, ..., 95–100 лет⟩, 100 лет},
- другой вариант для дат рождения – {1913 г.р., 1914 г.р., ..., 2021 г.р.},
- вариант для данных трекинговых систем – {01.01.2023 0:00–01.01.2023 0:10, 01.01.2023 10:00–01.01.2023 0:20, ..., 31.12.2023 23:50–01.01.2023 0:00},

• вариант для средств коммерции – {01.01.2023, 02.01.2023, ..., 31.12.2023}.

Далее в наборе НД(y) атрибут A_1, имевший тип дата/время, становится атрибутом, имеющим тип словарное значение.

3.2.5. Потоковые данные. Свойство повторяемости этого типа неизбежно приведет к наличию в наборе НД(y) множества однотипных атрибутов A_j, $j = 1,\ldots,m$, с собственно содержательными данными и атрибутов A_{m+l}, $l = 1,\ldots,m$, содержащих "привязки" данных A_j (временные и/или географические). При этом свойство быть квазиидентификатором для всей этой совокупности атрибутов в большей степени учтено при подготовке к обезличиванию исключением избыточных данных. Именно, для потоковых данных в наборе НД(y) выбран из потенциально неограниченного множества атрибутов A_j, $j = 1,\ldots,M$, с потоковой информацией небольшой набор A_j, $j = 1,\ldots,m$, $m \ll M$, из соображений уменьшения угрозы нарушения анонимности, в частности, такого, чтобы данные в атрибутах A_j были во всех элементах (или в большинстве) набора НД(y). Дальнейшая обработка потоковых данных сводится, таким образом, к определению типа и характеристики атрибутов A_j, $j = 1,\ldots,m$, с содержательной информацией, т.е. применением к ним одного из перечисленных выше шаблонов, а для атрибутов A_{m+l}, $l = 1,\ldots,m$, типа дата/время применяется шаблон разбиения. Если эти атрибуты содержат данные геопривязки, то к ним естественным будет применение шаблона синтеза числового непрерывного значения.

3.3. Оценка результатов обезличивания

Применение ко всем атрибутам набора НД(y) перечисленных шаблонов алгоритма обезличивания выполняется в обратном порядке – от шаблонов потоковых данных до универсальных шаблонов синтеза числовых величин. Полученный в результате набор ОД(y) содержит данные, которые следует интерпретировать как обезличенные. Будет ли этот набор итоговым, должна подтверждать оценка результата с точки зрения анонимности и полезности полученного результата. Для формирования такой оценки алгоритм обезличивания дополняется возможностями расчета некоторых числовых характеристик.

3.3.1. Объединение атрибутов. Корреляции. Важным вопросом, обязательным к рассмотрению при обезличивании данных, должно стать взаимное влияние значений разных атрибутов. Основные шаблоны алгоритма обезличивания, применяемые к группам атрибутов типов числовое дискретное значение и числовое непрерывное значение смогут сохранить характер зависимостей между значениями атрибутов в исходном наборе НД(y). Именно для этого формировались преобразования не отдельного атрибута A_1, а групп атрибутов $A_1,\ldots,A_m$. При этом надо отметить, что применение шаблонов алгоритма к группе в сравнении с последовательным применением этих же шаблонов к каждому отдельному атрибуту – более вычислительно затратный процесс. Эффективнее всего применять шаблоны к небольшим группам атрибутов, т.е. разбивать группу $A_1,\ldots,A_m$ на несколько небольших подгрупп, к каждой из которых уже применять соответствующий шаблон.

Для выявления групп зависимых атрибутов, а также для качественной оценки результатов обезличивания с точки зрения сохранения важных зависимостей, рекомендуется применение простого статистического метода оценки корреляций. Коэффициент корреляции корр_{12} значений двух атрибутов A_1 и A_2 определяется как

$$\text{корр}_{12}(A_1, A_2) = \frac{\sum\limits_{i=1}^{N}\langle A_{1i}\rangle\langle A_{2i}\rangle - \left(\sum\limits_{i=1}^{N}\langle A_{1i}\rangle\right)\left(\sum\limits_{i=1}^{N}\langle A_{2i}\rangle\right)}{\sqrt{\left(\sum\limits_{i=1}^{N}(\langle A_{1i}\rangle)^2 - \left(\sum\limits_{i=1}^{N}\langle A_{1i}\rangle\right)^2\right)\left(\sum\limits_{i=1}^{N}(\langle A_{2i}\rangle)^2 - \left(\sum\limits_{i=1}^{N}\langle A_{2i}\rangle\right)^2\right)}}.$$

Эта величина, принимающая значения от 0 до 1, характеризует степень линейной зависимости значений, принимаемых атрибутами A_1 и A_2 в наборе НД(y). Соответственно, при подготовке обезличивания можно вычислять коэффициент корреляции для любых пар атрибутов и принимать решение об отсутствии зависимостей (близости коэффициента к нулю) и относить атрибуты к разным подгруппам числовых типов, либо о ее наличии.

Аналогично вычисляются корреляции применительно к набору ОД(y) и, таким образом, есть возможность сравнения $\text{корр}_{12}(A_1, A_2)$ и $\text{корр}_{12}(\hat{A}_1, \hat{A}_2)$. Заметим, что ни один из методов обезличивания гарантированного сохранения корреляций не обеспечивает, поэтому анализ результирующего

набора ОД(y) таким инструментом вполне целесообразен.

3.3.2. Оценка уровня анонимности. Для анализа уровня анонимности в соответствии с данным в разделе 2.1 определением в наборе ОД(y) требуется:

• выбрать атрибуты $\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m$, участвующие в расчете,

• определить величины ε_j приемлемого малого отклонения значений атрибута $\hat{\mathrm{A}}_j$ для каждого j-го атрибута, принимающего числовые непрерывные значения,

• вычислить число K – гарантированную оценку уровня K-анонимности,

• вычислить число k – относительный уровень K-анонимности,

• вычислить число K_ε – гарантированную оценку уровня ε-анонимности.

Все эти вычисления имеет смысл также проводить на наборе НД(y) для соответствующих атрибутов $\mathrm{A}_1,\ldots,\mathrm{A}_m$.

Для величин ε_j для тех атрибутов $\hat{\mathrm{A}}_j$, что принимают числовые непрерывные значения, предлагается следующий метод. Задается уровень нечувствительности T в процентах, например $T = 10\%$ Среди значений $\langle\hat{\mathrm{A}}_j\rangle_i, i = 1,\ldots,N$, находится минимальное x_{min_j} и максимальное x_{max_j}, величина ε_j рассчитывается как

$$\varepsilon_j = \left(x_{max_j} - x_{min_j}\right)\frac{T}{200}.$$

Смысл этой величины таков. Если взять размах выборки $(x_{max_j} - x_{min_j})$ и рассмотреть случай, когда значения $\langle\hat{\mathrm{A}}_j\rangle_i$ атрибута $\hat{\mathrm{A}}_j$, представленные в наборе данных ОД(y), распределены максимально равномерно, то получится равномерная сетка с шагом $(x_{max_j} - x_{min_j})\frac{1}{N}$ (подразумевается, что поскольку тип значений атрибута числовое непрерывное значение, то в выборке нет повторяющихся значений, поэтому разных уникальных значений ровно N). Выбранное так значение ε_j направлено на то, чтобы в ε_j-окрестность каждого значения $\langle\hat{\mathrm{A}}_j\rangle_i$ попадало примерно $T\%$ имеющихся значений $\langle\hat{\mathrm{A}}_j\rangle_i$ (исключение составят крайние точки, близкие к x_{max_j} и x_{min_j}, при приближении к которым число точек в окрестности будет уменьшаться до $T/2$). Это "идеальное" распределение, в реальных данных будут точки, разбросанные как дальше, так и ближе друга от друга, чем в равномерной сетке. Соответственно, судить об уровне K_ε-анонимности можно, сравнивая разницу между идеальным значением T и фактической величиной K_ε.

Наконец надо отметить, что формального смысла анализировать уровень анонимности для атрибутов, обработанных шаблонами синтеза, вообще говоря, нет, т.к. данные – значения таких атрибутов – являются синтетическими, т.е. полностью анонимны и не могут в принципе рассматриваться в качестве угрозы нарушения анонимности, т.к. по своей природе не являются де-факто ПД.

3.3.3. Анализ полезности. Второй объект анализа результативности выполненного обезличивания – оценка полезности (применимости) данных результирующего набора ОД(y). Полезность результирующего набора ОД(y) предлагается определять разницей между наборами значений $\{\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$ и $\{\langle\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m\rangle_i\}_{i=1}^N$ в каждой из групп, выделенных в результате анализа корреляций. Поскольку эти наборы интерпретировались как реализации некоторых случайных векторов, то следует и различие между ними оценивать как различие между случайными векторами, т.е. соответствующими выборкам $\{\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$ и $\{\langle\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m\rangle_i\}_{i=1}^N$ вероятностными распределениями. Для определения количественной оценки разницы между распределениями значений атрибутов $\mathrm{A}_1,\ldots,\mathrm{A}_m$ в НД(y) и значений атрибутов $\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m$ в ОД(y) предлагается использовать типовое решение – расстояние Кульбака–Лейблера [22]. Его традиционно используют как числовую оценку меры сходства/расхождения между распределениями вероятностей двух случайных векторов (величин) ξ и $\hat{\xi}$.

Для определения расстояния Кульбака–Лейблера в случае атрибутов типа числовое дискретное значение сформируем из всех имеющихся данных $\{\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$ – значений атрибутов $\mathrm{A}_1,\ldots,\mathrm{A}_m$ в наборе НД(y) область значений $\{\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_l\}_{l=0}^{L-1}$ – все неповторяющиеся значения $\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i$, дополним их частотами p_l появления значения $\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_l$ в наборе $\{\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$ и частотами $\hat{p}_l$ появления значения $\langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_l$ в наборе $\{\langle\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m\rangle_i\}_{i=1}^N$, обозначим для простоты через $x_l = \langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_l$. Будем считать, что таким образом заданы величины ξ и $\hat{\xi}$ с одинаковой областью значений $\{x_l\}_{l=0}^{L-1}$ и рядами распределений $\{p_l\}_{l=0}^{L-1}$ и $\{\hat{p}_l\}_{l=0}^{L-1}$, соответственно, т.е.

$$p_l = \mathbf{P}\left(\xi = x_l = \langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_l\right),$$

$$\hat{p}_l = \mathbf{P}(\hat{\xi} = x_l = \langle\mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_l),$$

где $\mathbf{P}(\cdot)$ – вероятность события.

По предложенным обозначениям заметим следующее. Во-первых, строго говоря частоты

p_l, $\hat{p}_l$ нужно рассматривать как оценки неизвестных вероятностей, а не точные значения указанных вероятностей. В предположении, что выборка данных в НД(y) достаточно велика и статистически представительна (не содержит заведомых статистических искажений исходного распределения значений атрибутов), можно пренебречь погрешностью между частотной оценкой и истинной вероятностью. Во-вторых, будем учитывать, что величины $p_l > 0$ для всех $l = 0,\ldots, L-1$, по построению, потому что в область значений включены только те значения, что действительно имеются в наборе $\{\langle \mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$. При этом среди величин $\hat{p}_l$ могут оказаться нулевые в силу случайного характера примененного шаблона обезличивания.

Мерой полезности распределения $\{(x_l, \hat{p}_l)\}_{l=0}^{L-1}$ по отношению к распределению $\{(x_l, p_l)\}_{l=0}^{L-1}$ называется расстояние Кульбака–Лейблера $D(\hat{\xi},\xi)$:

$$D(\hat{\xi},\xi) = \sum_{l=0}^{L-1} \hat{p}_l \ln\left(\frac{\hat{p}_l}{p_l}\right).$$

Заметим, что величина $D(\hat{\xi},\xi)$ является несимметричной ($\hat{\xi},\xi$ нельзя поменять местами), в рассматриваемой постановке существует (может быть вычислена) из-за определения величин p_l (их положительность гарантирует существование логарифма). Известно, что значение $D(\hat{\xi},\xi) \geq 0$, причем равенство $D(\hat{\xi},\xi) = D(\xi,\hat{\xi})$ означает, что $\hat{p}_l = p_l$ для всех $l = 0,\ldots, L-1$. Масштабной характеристики (большой-малый) величина $D(\hat{\xi},\xi)$ не имеет, так что степень близости к 0 определяется экспериментально.

В случае числового непрерывного значения имеющиеся данные двух наборов – значения $\{\langle \mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$ атрибутов $\mathrm{A}_1,\ldots,\mathrm{A}_m$ в наборе НД(y) и значения $\{\langle \hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m\rangle_i\}_{i=1}^N$ атрибутов $\hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m$ в наборе ОД(y) интерпретируются как выборки из распределений случайных векторов $\xi = \mathrm{col}(\xi_1,\ldots,\xi_m)$ и $\hat{\xi} = \mathrm{col}(\hat{\xi}_1,\ldots,\hat{\xi}_m)$, каждый элемент которых предполагается непрерывной случайной величиной. Таким образом, законы распределения ξ и $\hat{\xi}$ определяются плотностями вероятности – функциями $f_\xi(x_1,\ldots,x_m)$ и $f_{\hat{\xi}}(x_1,\ldots,x_m)$ соответственно:

$$\mathbf{P}(\xi \in X) = \int_X f_\xi(x_1, \ldots, x_m)dx_1 \ldots dx_m,$$
$$\mathbf{P}(\hat{\xi} \in X) = \int_X f_{\hat{\xi}}(x_1, \ldots, x_m)dx_1 \ldots dx_m.$$

Поскольку истинные распределения f_ξ и $f_{\hat{\xi}}$, вообще говоря, неизвестны, то в качестве плотностей будут использоваться, как и в случае числового дискретного значения, их ядерные оценки.

Именно, вычислим величины $\mu = \mathrm{col}(\mu_1, \ldots, \mu_n)$, $\sigma = \mathrm{diag}(\sigma_1,\ldots,\sigma_m)$, $\hat{\mu} = \mathrm{col}(\hat{\mu}_1,\ldots,\hat{\mu}_m)$, $\hat{\sigma} = \mathrm{diag}(\hat{\sigma}_1,\ldots,\hat{\sigma}_m)$, т.е. выборочные средние значения и среднеквадратические отклонения имеющихся выборок $\{\langle \mathrm{A}_1,\ldots,\mathrm{A}_m\rangle_i\}_{i=1}^N$ и $\{\langle \hat{\mathrm{A}}_1,\ldots,\hat{\mathrm{A}}_m\rangle_i\}_{i=1}^N$. В дополнение к определенной выше шаблоном синтеза числового непрерывного значения плотности $f_\xi(x_1, \ldots, x_m)$, положим

$$\hat{f}_\xi(x_1, \ldots, x_m) = \frac{1}{N\sqrt{\det(\hat{H}^T\hat{H})}} \times$$
$$\times \sum_{i=1}^N \Phi(x_1, \ldots, x_m; \mathrm{col}(\langle \hat{\mathrm{A}}_1, \ldots, \hat{\mathrm{A}}_m\rangle_i), \hat{H}^2),$$
$$\hat{H}^2 = \mathrm{diag}(\hat{h}_1^2,\ldots,\hat{h}_m^2), \quad \hat{h}_j = \sqrt[m+4]{\left(\frac{4}{m+2}\right)}\frac{1}{\sqrt[m+4]{N}}\hat{\sigma}_j.$$

Мерой полезности распределения $f_\xi(x_1,\ldots,x_m)$ по отношению к распределению $f_\xi(x_1,\ldots,x_m)$ называется расстояние Кульбака–Лейблера $D(\hat{f}_\xi, \hat{f}_{\hat{\xi}})$ распределений $\hat{f}_\xi(x_1,\ldots,x_m)$ и $\hat{f}_{\hat{\xi}}(x_1,\ldots,x_m)$:

$$D(\hat{f}_\xi, f_{\hat{\xi}}) = \int_{-\infty}^{+\infty} \ldots \int_{-\infty}^{+\infty} \hat{f}_\xi(x_1,\ldots,x_m) \times$$
$$\times \ln\left(\frac{\hat{f}_\xi(x_1,\ldots,x_m)}{f_\xi(x_1,\ldots,x_m)}\right)dx_1 \ldots dx_m.$$

Величина $D(\hat{f}_\xi, f_{\hat{\xi}})$ имеет те же свойства, что и величина $D(\hat{\xi},\xi)$ полезности для типа числовое дискретное значение.

4. ПРАКТИЧЕСКИЙ ПРИМЕР ОБЕЗЛИЧИВАНИЯ

4.1. Входные и выходные данные

Для выполнения модельных расчетов с предложенным алгоритмом требуется прежде всего решить вопрос с источником исходных необезличенных данных. Поскольку реальные данные в исследовательских целях применяться не могут, нужно получить какие-либо макетные данные, имитирующие реальные ПД. С этой целью была создана простая база данных по теме гражданских авиационных перевозок. Идея для создания набора ОД, имеющего практический смысл, состояла в использовании открытых данных по авиаперевозкам, выполненным в РФ в 2019 году. Во-первых, мы использовали действующее на тот момент расписание, которое позволило сформиро-

вать служебные конструкции – списки аэропортов и авиакомпаний. Кроме того, мы получили типы воздушных судов для большей части рейсов, что дало возможность имитировать число перевезенных пассажиров, исходя из средней загруженности рейсов 75–85%. Для пассажиров имитировался перелет туда и обратно по имеющемуся расписанию (по этой причине достаточно только информации об аэропорте отправления), и еще случайным образом задавался возраст (для этого использовалось дискретное распределение, полученное из гауссовского). В итоге в базу данных включены следующие сведения:

1) синтетический идентификатор пассажира,

2) возраст (лет),

3) дата-время отправления,

4) дата-время прибытия,

5) аэропорт отправления,

6) авиакомпания,

7) атрибут отправления,

8) атрибут прибытия.

Последние два атрибута включены для имитации атрибутов ПД типа числовое непрерывное значение и получены следующим образом. “Атрибут отправления” является производным значения “дата-время отправления”: тип дата/время преобразован в тип вещественный и зашумлен случайным значением, равномерно распределенным на отрезке [–1, 1] (это очень маленький шум, единственная задача которого убрать прямую зависимость этих двух атрибутов, имеющую место из-за того, что фактическая дата-время отправления однозначно определяет дату-время прибытия). Аналогично “атрибут прибытия” получается из “дата-время прибытия”. Таким образом, в НД есть два атрибута непрерывного типа, и они зависимы друг от друга.

Далее, к атрибуту “возраст (лет)” применялся описанный выше шаблон разбиения. Результирующий атрибут “возраст (группа)” вместе с атрибутами “аэропорт отправления”, “авиакомпания” дал три атрибута дискретного типа, и они независимы друг от друга. Независимость очевидна для возраста, поскольку его значения моделировались независимо. Для аэропортов и авиакомпаний независимость (точнее отсутствие выраженной линейной зависимости) объясняется тем, что в выборку всегда входит пара прямой и обратный рейс. Это объяснение подтверждается расчетами.

Итого в ОД вошли: синтетический идентификатор пассажира, возраст (группа), аэропорт отправления, авиакомпания, атрибут отправления, атрибут прибытия.

4.2. Результаты расчетов

Для 10000 записей о перелетах в наборе НД в результате применения алгоритма обезличивания формировался набор ОД такого же размера. Для анализа результатов вначале рассмотрим корреляционные показатели для двух групп атрибутов – дискретных и непрерывных. Для группы дискретных атрибутов корреляции $\text{корр}(A_i, A_j)$ в наборах:

$$\text{НД: корр (возраст, аэропорт)} = 0.021,$$

$$\text{корр(возраст, авиакомпания)} = 0.077,$$

$$\text{корр(аэропорт, авиакомпания)} = 0.03,$$

$$\text{ОД: корр(возраст, аэропорт)} = 0.059,$$

$$\text{корр(возраст, авиакомпания)} = 0.06,$$

$$\text{корр(аэропорт, авиакомпания)} = 0.029.$$

Для группы непрерывных атрибутов корреляции в наборах:

$$\text{НД: корр (атрибут отправления, атрибут прибытия)} = 0.966,$$

$$\text{ОД: корр (атрибут отправления, атрибут прибытия)} = 0.931.$$

Отметим, что эти результаты нужны не только для того, чтобы убедиться в работоспособности частотных и ядерных оценок при моделировании выборок случайных значений, что и так очевидно, но и для анализа влияния на эти показатели шаблона разбиения, который применялся к атрибуту возраст.

Другой показатель дает введенная в предыдущем разделе мера полезности (полезн(A_i, ..., A_j)). Поскольку масштаб этой меры заранее определить нельзя, то для сравнения эксперимент по обезличиванию был повторен с заменой метода синтеза на метод зашумления. Вот некоторые результаты:

для метода синтеза:

$$\text{полезн(атрибут отправления)} = 0.396,$$

$$\text{полезн (атрибут отправления, атрибут прибытия)} = 0.315,$$

$$\text{полезн (возраст, авиакомпания, аэропорт, атрибут отправления, атрибут прибытия)} = 0.129,$$

для метода зашумелния

$$\text{полезн (атрибут прибытия)} = 0.537,$$

$$\text{полезн (атрибут отправления, атрибут прибытия)} = 0.427,$$

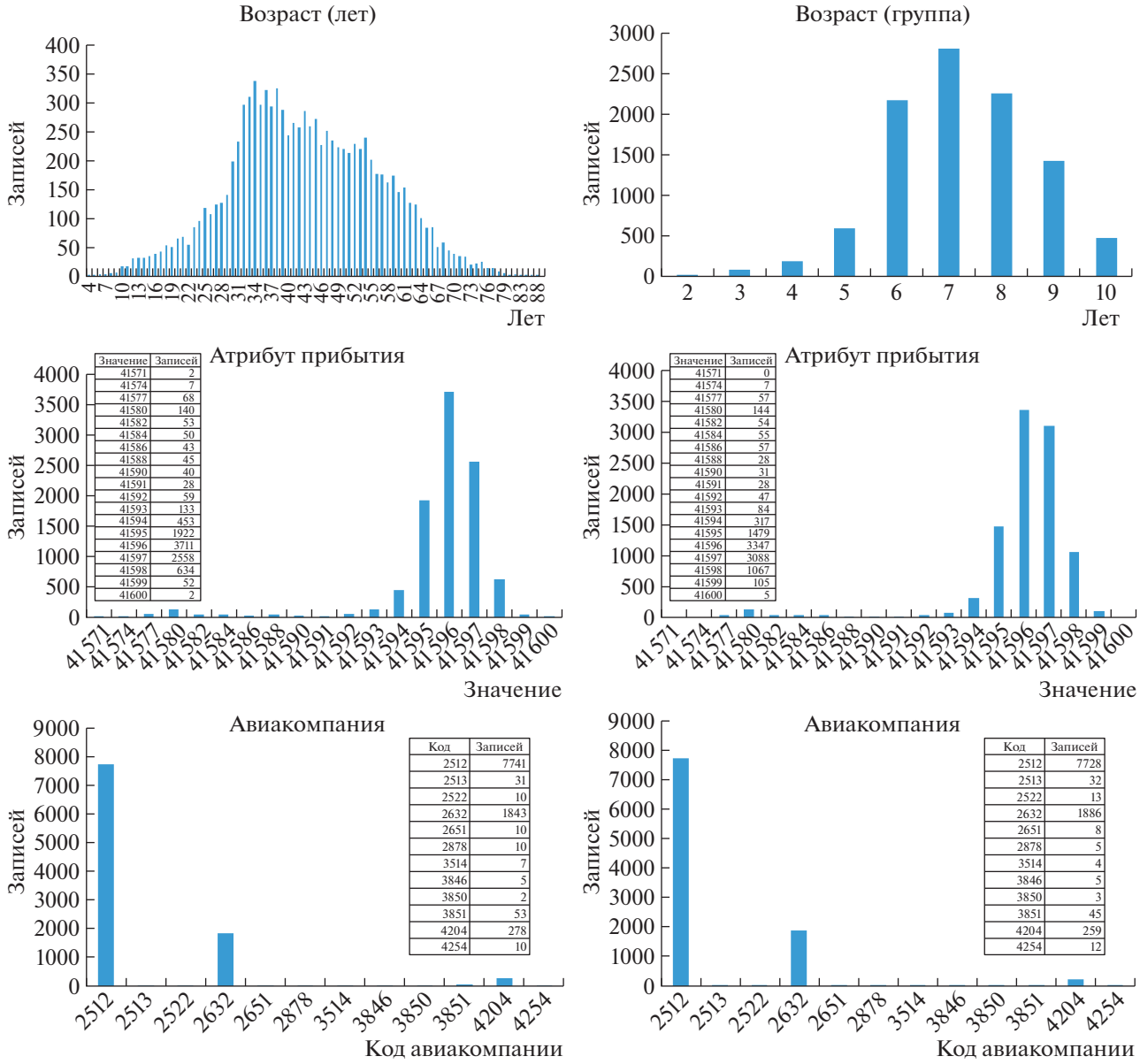

Рис. 2. Пример гистограмм необезличенных и обезличенных атрибутов.

полезн(возраст, авиакомпания, аэропорт,
атрибут отправления,
атрибут прибытия) = 1.725.

Для визуальной оценки этих результатов можно использовать гистограммы распределений. Примеры для трех атрибутов представлены парами (необезличенный-обезличенный) на рис. 2.

Наконец, нужно сделать замечание об уровне анонимности. Непрерывные атрибуты, использованные для данного примера, были выбраны так, чтобы принципиально усложнить реализацию концепции K-анонимности. Оставив атрибуты “дата-время отправления” и “дата-время прибытия” неизменными, об уровне K-анонимности можно было бы говорить вполне предметно. Изменение этих атрибутов с помощью малого непрерывного шума привело к тому, что уровень K-анонимности в обоих наборах данных стал равным 1. Более того, малость шума приводит к тому, что и уровень K_ε-анонимности столь же мал. Именно, в НД он также равен 1, в ОД значение вырастает до 3, что конечно же существенно ситуацию не меняет. Не столь ожидаемая, но такая же неудачная для K-анонимности ситуация оказалась и для дискретных атрибутов. Получилось, что при использованном числе записей и распределении пассажиров по возрасту есть возрастные

группы младенцев, в которых на некоторые авиакомпании попадает по одному рейсу для пассажира такого возраста. И ситуация воспроизводится при синтезе. Так что и здесь уровень *K*-анонимности оказывается равным 1. Эта неприятность могла бы не возникнуть, если бы был применен шаблон устранения аномалий, как и предусмотрено алгоритмом обезличивания. Не сделав этого, мы еще больше подчеркнули преимущество концепции синтеза.

Для целей данной работы это хороший пример, потому что для полученного результата, для данных, обезличенных предложенным методом синтеза, расчет уровня *K*-анонимности, строго говоря, ничего не означает. Все синтезированные записи набора ОД анонимны на 100%, потому что в принципе нет субъекта ПД для этих записей из-за их полностью синтетического характера.

5. ЗАКЛЮЧЕНИЕ

В этой и предыдущей статьях был охвачен довольно широкий спектр вопросов по теме автоматизации обезличивания персональных данных. Внимание к тематике ПД в нашей стране столь же велико, как и во всех государствах, обладающих значительной информационной инфраструктурой. Но это внимание ограничено нетехническими сообществами. Лучше всего проработаны правовые вопросы, на хорошем уровне находится нормативная база, предметно работают уполномоченные госорганы, ведется социальная дискуссия. Но при этом значимое внимание со стороны научного сообщества, исследовательская активность отсутствуют. В двух статьях была сделана попытка показать широту и разнообразие именно технических проблем и задач в данной области и включиться в исследовательскую активность со своими идеями.

6. БЛАГОДАРНОСТИ

Работа выполнялась с использованием инфраструктуры Центра коллективного пользования “Высокопроизводительные вычисления и большие данные” (ЦКП “Информатика” ФИЦ ИУ РАН, Москва).

СПИСОК ЛИТЕРАТУРЫ

1. *Борисов А.В., Босов А.В., Иванов А.В.* Применение имитационного компьютерного моделирования к задаче обезличивания персональных данных. Оценка состояния и основные положения // Программирование, 2023. № 4, с. 58–74.
2. *Aggarwal C.C., Yu P.S.* On Privacy-Preservation of Text and Sparse Binary Data with Sketches // SIAM Conference on Data Mining, 2007.
3. *Sweeney L. K*-anonymity: a model for protecting privacy // International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 2002. V. 10. № 5. P. 557–570.
4. *Samarati P., Sweeney L.* Generalizing Data to Provide Anonymity when Disclosing Information (Abstract) // Proc. of ACM Symposium on Principles of Database Systems, 1998. P. 188.
5. *Samarati P.* Protecting Respondents’ Identities in Microdata Release // IEEE Trans. Knowl. Data Eng., 2001. V. 13. № 6. P. 1010–1027.
6. *Bayardo R.J., Agrawal R.* Data Privacy through Optimal k-Anonymization // Proceedings of the ICDE Conference, 2005. P. 217–228.
7. *Fung B., Wang K., Yu P.* Top-Down Specialization for Information and Privacy Preservation // ICDE Conference, 2005.
8. *Wang K., Yu P., Chakraborty S.* Bottom-Up Generalization: A Data Mining Solution to Privacy Protection // ICDM Conference, 2004.
9. *Domingo-Ferrer J., Mateo-Sanz J.* Practical data-oriented micro-aggregation for statistical disclosure control // IEEE TKDE, 2002. V. 14. № 1.
10. *Winkler W.* Using simulated annealing for k-anonymity // Technical Report 7, US Census Bureau, Washington D.C. 20233, 2002.
11. *Iyengar V.S.* Transforming Data to Satisfy Privacy Constraints // KDD Conference, 2002.
12. *Lakshmanan L., Ng R., Ramesh G.* To Do or Not To Do: The Dilemma of Disclosing Anonymized Data // ACM SIGMOD Conference, 2005.
13. *Aggarwal C.C., Yu P.S.* On Variable Constraints in Privacy-Preserving Data Mining // SIAM Conference, 2005.
14. *Aggarwal C.C.* On k-anonymity and the curse of dimensionality // VLDB Conference, 2005.
15. *Iyengar V.S.* Transforming Data to Satisfy Privacy Constraints // KDD Conference, 2002.
16. *Machanavajjhala A., Gehrke J., Kifer D., Venkitasubramaniam M. L*-Diversity: Privacy Beyond k-Anonymity // ICDE Conference, 2006.
17. *Fung B., Wang K., Yu P.* Top-Down Specialization for Information and Privacy Preservation // ICDE Conference, 2005.
18. *Wang K., Yu P., Chakraborty S.* Bottom-Up Generalization: A Data Mining Solution to Privacy Protection // ICDM Conference, 2004.
19. *Rosenblatt M.* Remarks on Some Nonparametric Estimates of a Density Function // Ann. Math. Statist., 1956. V. 27. № 3. P. 832–837.
20. *Parzen E.* On Estimation of a Probability Density Function and Mode // Ann. Math. Statist., 1962. V. 33. № 3. P. 1065–1076.
21. *Silverman B.W.* Density Estimation for Statistics and Data Analysis. London: Chapman & Hall/CRC, 1986.
22. *Kullback S., Leibler R.A.* On information and sufficiency // Ann. Math. Statist., 1951. V. 22. № 1. P. 79–86.

APPLICATION OF SIMULATED COMPUTER SIMULATION TO THE TASK OF PERSONAL DEPERSONALIZATION DATA. MODEL AND ALGORITHM FOR DECONTAMINATION BY SYNTHESIS

© 2023 г. S. A. Borisov[a,#], A. A. Bosov[a,##], and D. E. Ivanov[a,###]

[a] *Federal Research Center "Informatics and Management" RAS, Moscow, Russia*

[#] *e-mail: aborisov@ipiran.ru*

[##] *e-mail: avbosov@ipiran.ru*

[###] *e-mail: aivanov@ipiran.ru*

The second part of the study on the topic of automated depersonalization of personal data is presented. The review and analysis of the prospects for research, performed earlier, is supplemented here by a practical result. A model of the depersonalization process is proposed, reducing task of ensuring anonymity of personal data to manipulation of samples of different types of random elements. Accordingly, the key idea of transforming data to ensure their anonymity, provided that utility is maintained, is to apply the synthesis method, i.e. complete replacement of all unpublished data with synthetic values. The proposed model identifies a set of element types for which synthesis patterns are proposed. The set of patterns compiles the depersonalization algorithm by the synthesis method. Methodically, each template is based on a typical statistical tool – frequency probability estimates, nuclear Rosenblatt-Parsen density estimates, statistical averages and covariances. The application of the algorithm is illustrated by a simple example from the field of civil air transportation.

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 35–46

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 004.421.6

ПОИСК ЛОРАНОВЫХ РЕШЕНИЙ СИСТЕМ ЛИНЕЙНЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ С УСЕЧЕННЫМИ СТЕПЕННЫМИ РЯДАМИ В РОЛИ КОЭФФИЦИЕНТОВ

© 2023 г. С. А. Абрамов[a],*, А. А. Рябенко[a],, Д. Е. Хмельнов[a],*****

[a]*Федеральный исследовательский центр "Информатика и управление" РАН*
119333 Москва, ул. Вавилова, 40, Россия
**E-mail: sergeyabramov@mail.ru*
***E-mail: anna.ryabenko@gmail.com*
****E-mail: dennis_khmelnov@mail.ru*

Поступила в редакцию 31.08.2022 г.
После доработки 16.10.2022 г.
Принята к публикации 30.10.2022 г.

Рассматриваются системы линейных обыкновенных дифференциальных уравнений с бесконечными формальными степенными рядами в роли коэффициентов. Ряды задаются в усеченном виде, при этом степени усечения могут различаться для разных коэффициентов. В качестве средства исследования таких систем привлекаются индуцированные рекуррентные системы и литеральные обозначения незаданных коэффициентов рядов. Для случая, когда определитель ведущей матрицы индуцированной системы отличен от нуля и не содержит литералов, предлагается алгоритм построения лорановых решений системы. Ряды, входящие в решения, вновь являются усеченными. Алгоритм находит для них максимально возможное число членов, инвариантных относительно любых продолжений усеченных коэффициентов исходной системы. Представлены реализация алгоритма в виде Maple-процедуры и примеры ее использования.

DOI: 10.31857/S0132347423020036, **EDN:** GYMMYQ

1. ВВЕДЕНИЕ

В [1–3] были предложены алгоритмы поиска решений систем линейных обыкновенных дифференциальных уравнений с бесконечными формальными степенными рядами в роли коэффициентов, при этом степенные ряды были заданы алгоритмически, т.е. предполагался известный алгоритм, вычисляющий по значению n коэффициент при x^n. Ниже ряды задаются в усеченном виде, что означает, что мы не располагаем полной информацией об уравнениях системы. В [4–6] нами предложены алгоритмы поиска различных видов решений скалярных (одиночных) линейных обыкновенных дифференциальных уравнений с коэффициентами в виде усеченных степенных рядов. Исходя из такой неполной информации об уравнении, алгоритмы из [4–6] дают максимально возможное число членов, входящих в решения, таких, что эти члены инвариантны относительно всех возможных продолжений коэффициентов заданного уравнения. В данной работе рассмотрена аналогичная задача поиска лорановых решений систем линейных обыкновенных дифференциальных уравнений с усеченными степенными рядами в роли коэффициентов. Переход к случаю систем имеет специфические особенности, которые рассмотрены далее и учтены в предлагаемом алгоритме.

Задачи этого рода рассматривались ранее в [7, 8] для систем произвольного порядка

$$A_r(x)\theta^r y(x) + A_{r-1}(x)\theta^{r-1} y(x) + \cdots + A_0(x)y(x) = 0, \quad (1)$$

где $y(x) = \left(y_1(x), y_2(x), \ldots, y_m(x)\right)^T$ – вектор неизвестных, $A_r(x), \ldots, A_0(x)$ – $m \times m$-матрицы с элементами в виде усеченных рядов, $\theta = x\dfrac{d}{dx}$. В [7] предполагается, что возможно получение усечения любой степени для рядов в $A_r(x), \ldots, A_0(x)$, и определяется, какой степени усечения этих рядов достаточно, чтобы вычислить лорановы решения требуемой степени усечения. В [8] матрицы $A_r(x)$, ..., $A_0(x)$ рассматриваются как полиномиальные, представляющие собой усечения рядов одной степени усечения, при этом рассматривается случай, когда матрица $A_r(x)$ является *строго невырожденной*, т.е. она не только невырождена, но и любое продолжение ее элементов посредством

добавления членов старших степеней до бесконечных рядов оставляет ее невырожденной. Ниже мы рассматриваем случай, когда выписанная для системы (1) индуцированная рекуррентная система (см. определение в разд. 2.3) имеет невырожденную ведущую матрицу и ее определитель не содержит литералов (см. определение в разд. 4.1). Здесь подчеркнем, что такая невырожденность в общем случае не влечет невырожденность $A_r(x)$ в (1). Также отметим, что в данной работе, в отличие от [7, 8], степени усечения различных рядов в $A_r(x),\ldots,A_0(x)$ могут быть различны, как и степени усечения компонент усеченного лоранова решения – для каждого ряда находится максимально возможное число членов.

2. ПРЕДВАРИТЕЛЬНЫЕ СВЕДЕНИЯ

2.1. Системы, усеченные ряды и продолжения

Пусть K – алгебраически замкнутое числовое поле. Для кольца полиномов от x над K мы в дальнейшем используем обычное обозначение $K[x]$. Кольцо формальных степенных рядов от x над K обозначается через $K[[x]]$, поле формальных лорановых рядов – через $K((x))$. Для ненулевого элемента $a(x)=\sum a_i x^i$ из $K((x))$ его *валюация* $\mathrm{val}_x a(x)$ определена равенством $\mathrm{val}_x a(x)=\min\{i \mid a_i \neq 0\}$, при этом $\mathrm{val}_x 0=\infty$. Валюация вектора или матрицы с компонентами-рядами считается равной минимуму валюаций компонент. Пусть $t\in\mathbb{Z}$, *t-усечение* $a^{\langle t\rangle}(x)$ получается отбрасыванием всех членов ряда $a(x)$ степени большей, чем t. Число t называется *степенью усечения* ряда. Под $O(x^{t+1})$, где $t\in\mathbb{Z}$, будет пониматься некий (неуточняемый) ряд с валюацией, большей t. Как правило, это обозначение используется в тех случаях, когда либо подразумеваемый ряд нам неизвестен, либо если конкретный вид ряда не представляет для нас интереса; важно лишь, что его валюация больше, чем t.

Если R – некоторое кольцо, то $\mathrm{Mat}_m(R)$ обозначает кольцо $m\times m$-матриц с элементами из R.

Дифференциальные системы будет удобно записывать с помощью операции $\theta = x\dfrac{d}{dx}$ вместо обычной операции дифференцирования $\dfrac{d}{dx}$ (переход от одной формы записи к другой выполняется легко). Мы рассматриваем системы вида (1). Относительно

$$A_0(x), A_1(x),\ldots,A_r(x)$$

предполагается, что $A_k(x)\in \mathrm{Mat}_m(K[[x]])$, $k=0,1,\ldots,r$, при этом *ведущая* матрица $A_r(x)$ системы является ненулевой. Также предполагается, что валюация по крайней мере одной из матриц $A_0(x)$, $A_1(x),\ldots,A_r(x)$ равна нулю, т.е. валюация системы (1) равна нулю.

Элементы матриц $A_k(x)$ будем называть *коэффициентами системы*, ими будут у нас формальные степенные ряды. Элемент матрицы $A_k(x)$ на пересечении i-й строки и j-го столбца будем обозначать через $A_{kij}(x)$. В случае системы, коэффициенты которой заданы в усеченном виде, известны такие неотрицательные целые $t_{0ij}, t_{1ij},\ldots,t_{rij}$, что

$$A_{kij}(x) = a_{kij}(x) + O(x^{t_{kij}+1}), \qquad (2)$$

где $a_{kij}(x)\in K[x]$, $t_{kij}\geq \deg a_{kij}(x)$, $k=0,1,\ldots,r$, $i=1,\ldots,m$, $j=1,\ldots,m$.

Продолжение усеченного ряда – это ряд, возможно также усеченный, начальные члены которого совпадают с известными начальными членами исходного усеченного ряда; соответственно, продолжением усеченного уравнения назовем уравнение, коэффициенты которого являются продолжениями коэффициентов исходного уравнения, а продолжением системы уравнений назовем систему, уравнения которой являются продолжениями уравнений исходной системы.

Пусть (1) состоит из линейно независимых над $K[[x]][\theta]$ уравнений. В этом случае говорим, что система имеет *полный ранг*, саму систему называем системой *полного ранга*. Из [7], предл. 2 следует, что для случая систем, коэффициенты которых являются рядами, заданными алгоритмически, задача проверки независимости уравнений над $K[[x]][\theta]$ является алгоритмически неразрешимой. Предложенные в [1–3] алгоритмы предполагают, что система имеет полный ранг. Для системы с усеченными коэффициентами будем предполагать, что любое ее продолжение является системой полного ранга.

2.2. Лорановы решения систем

Решение $y(x)=(y_1(x),\ldots,y_m(x))^T$ дифференциальной системы (1), компоненты $y_i(x)$ которого являются формальными лорановыми рядами, называется *лорановым*. Для системы полного ранга, коэффициенты которой являются рядами, заданными алгоритмически, алгоритм из [1] находит все ее усеченные лорановы решения с любой заданной степенью усечения. Для усеченной системы нет возможности вычислять решения с произвольной степенью усечения. В [4] это было доказано для частного случая – скалярного уравнения ($m=1$). Ниже в разд. 3 будет определено, какими именно усеченными лорановыми решениями мы занимаемся в случае усеченных систем ($m>1$).

2.3. Индуцированные рекуррентные системы

Пусть система (1) имеет коэффициенты, которые являются алгоритмически заданными рядами. Пусть также ее лораново решение имеет вид

$$y(x) = \sum_{n=v}^{\infty} u(n)x^n, \tag{3}$$

где $v \in \mathbb{Z}$, $u(n) = (u_1(n),\ldots,u_m(n))^T \in K^m$ для $n \in \mathbb{Z}$. Тогда оказывается (см. [1]), что последовательность векторов $u(n)$ удовлетворяет *индуцированной рекуррентной* системе $R(u) = 0$. Эта система строится преобразованием

$$x \to E^{-1}, \quad \theta \to n, \tag{4}$$

применяемым к исходной дифференциальной системе (1), здесь E^{-1} обозначает оператор сдвига:

$$E^{-1}u(n) = u(n-1).$$

Имеем

$$R = B_0(n) + B_{-1}(n)E^{-1} + B_{-2}(n)E^{-2} + \ldots$$

Индуцированная система запишется в виде

$$B_0(n)u(n) + B_{-1}(n)u(n-1) + \ldots = 0, \tag{5}$$

где

• $u(n) = (u_1(n),\ldots,u_m(n))^T$ – вектор-столбец неизвестных последовательностей таких, что $u_i(n) = 0$ для всех отрицательных n с достаточно большим значением $|n|$, $i = 1,\ldots,m$;

• $B_0(n), B_{-1}(n),\ldots \in \mathrm{M}at_m(K[n])$;

• $B_0(n)$ – ведущая матрица системы (5) (это ненулевая матрица, поскольку валюация системы (1) равна нулю).

Очевидно, что индуцированная система (5) имеет полный ранг, т.е. ее уравнения независимы над $K[n][[E^{-1}]]$, если и только если исходная дифференциальная система (1) имеет полный ранг.

Если матрица $B_0(n)$ невырождена, то $\det B_0(n) = 0$ может рассматриваться как определяющее уравнение исходной дифференциальной системы: множество целых корней этого алгебраического уравнения включает множество всех возможных валюаций лорановых решений системы (1). Это позволяет, в частности, найти нижнюю границу валюаций всех лорановых решений системы. Если $\det B_0(n) = 0$ не имеет целых корней, система не имеет лорановых решений.

3. ПОСТАНОВКА ЗАДАЧИ

Рассматриваемая нами задача состоит в том, чтобы для заданной системы (1), коэффициенты которой являются усеченными рядами вида (2), найти в компонентах $y_i(x)$ усеченных лорановых решений максимально возможное число начальных членов, инвариантных относительно любых продолжений этой системы. При этом будем интересоваться такими усеченными решениями, в которых хотя бы в одной компоненте есть хотя бы один ненулевой член.

Инвариантность членов усеченного решения $y(x)$ означает, что множество лорановых решений (возможно, также усеченных) любого продолжения заданной системы будет содержать решение, начальные члены в компонентах которого совпадают со всеми членами в компонентах $y(x)$. Максимальность же числа членов в решении $y(x)$ означает, что продолжение хотя бы на одно слагаемое хотя бы одной компоненты решения приведет к нарушению инвариантности, т.е. будет существовать такое продолжение заданной системы, у которого нет решения, начальные члены в компонентах которого совпадают с таким продолжением решения.

Для системы (1) множество W всех усеченных лорановых решений с максимальной степенью усечения будем представлять набором $\bar{W} = [w_1,\ldots,w_p]$ вектор-столбцов, компоненты которых являются усеченными лорановыми рядами с произвольными постоянными в коэффициентах. Каждый из столбцов имеет вид

$$w_i = \begin{pmatrix} \sum_{n=v_{i1}}^{q_{i1}} C_{i1n}x^n + O(x^{q_{i1}+1}) \\ \ldots \\ \sum_{n=v_{im}}^{q_{im}} C_{imn}x^n + O(x^{q_{im}+1}) \end{pmatrix}$$

где для $i = 1,\ldots,p$, $j = 1,\ldots,m$, $n = v_{ij},\ldots,q_{ij}$ коэффициент C_{ijn} является линейной комбинацией произвольных постоянных, которые могут принимать все такие значения из K, что $C_{ijv_j} \neq 0$. Таким образом v_{ij} – валюация соответствующего ряда, q_{ij} – его степень усечения. Валюации компонент разных векторов в наборе $\bar{W}$ различаются хотя бы для одной компоненты: для любых i_1, i_2 $(i_1 \neq i_2)$ выполняется $(v_{i_1,1},\ldots,v_{i_1,m}) \neq (v_{i_2,1},\ldots,v_{i_2,m})$. Т.е. множество W усеченных решений разбито на подмножества с различными комбинациями валюаций компонент. Для одних и тех же компонент разных векторов набора $\bar{W}$ степень усечения может быть различной. Любому усеченному решению системы (1) соответствуют некоторые определенные значения произвольных постоянных в одном из векторов набора $\bar{W}$.

В качестве иллюстрации рассмотрим систему, представленную на рис. 1. Для этой системы мно-

$$\begin{pmatrix} O(x^3) & O(x^3) & O(x^3) \\ O(x^3) & -1+x+x^2+O(x^3) & O(x^3) \\ O(x^3) & -1+x+x^2+O(x^3) & O(x^3) \end{pmatrix}\theta^2 y(x) + \begin{pmatrix} 1+x+O(x^2) & O(x^3) & O(x^3) \\ O(x^3) & 2-4x-4x^2+O(x^3) & O(x^3) \\ 1+x+O(x^2) & 2-4x-4x^2+O(x^3) & 1+O(x^5) \end{pmatrix}\theta y(x) +$$
$$+\begin{pmatrix} -1+\frac{1}{2}x^2+O(x^3) & O(x^3) & O(x^3) \\ O(x^3) & O(x^6) & O(x^3) \\ -1-x+\frac{1}{2}x^2+O(x^3) & -1+O(x^4) & O(x^3) \end{pmatrix} y(x) = 0$$

Рис. 1.

$$\left[\begin{pmatrix} c_1x-c_1x^2+O(x^3) \\ c_2x^2+O(x^3) \\ c_3+\left(\frac{c_1}{2}+\frac{c_2}{2}\right)x^2+O(x^3) \end{pmatrix}, \begin{pmatrix} c_1x-c_1x^2+O(x^3) \\ c_2x^2-\frac{4c_2}{3}x^3+O(x^4) \\ \left(\frac{c_1}{2}+\frac{c_2}{2}\right)x^2+O(x^3) \end{pmatrix}, \begin{pmatrix} -c_2x+c_2x^2+O(x^3) \\ c_2x^2-\frac{4c_2}{3}x^3+O(x^4) \\ O(x^3) \end{pmatrix}, \begin{pmatrix} c_1x-c_1x^2+O(x^3) \\ O(x^3) \\ c_3+\frac{c_1}{2}x^2+O(x^3) \end{pmatrix},\right.$$
$$\left.\begin{pmatrix} c_1x-c_1x^2+O(x^3) \\ O(x^4) \\ \frac{c_1}{2}x^2+O(x^3) \end{pmatrix}, \begin{pmatrix} O(x^3) \\ c_2x^2+O(x^3) \\ c_3+\frac{c_2}{2}x^2+O(x^3) \end{pmatrix}, \begin{pmatrix} O(x^5) \\ c_2x^2-\frac{4c_2}{3}x^3+O(x^5) \\ \frac{c_2}{2}x^2-\frac{4c_2}{9}x^3+O(x^5) \end{pmatrix}, \begin{pmatrix} O(x^3) \\ O(x^3) \\ c_3+O(x^3) \end{pmatrix}\right]$$

Рис. 2.

жество лорановых решений с максимальной степенью усечения задается набором из восьми вектор-столбцов, представленным на рис. 2 (здесь и далее c_i обозначает произвольную постоянную, i – натуральное число – индекс этой произвольной постоянной). Несложно видеть, что комбинации валюаций компонент векторов различны. Также видно, что степени усечения компонент векторов могут различаться: у первого, четвертого, шестого и восьмого векторов степень усечения всех компонент равна 2, у седьмого вектора степень усечения всех компонент равна 4, у второго, третьего и пятого векторов степень усечения первой и третьей компоненты равна 2, второй компоненты равна 3. Таким образом в данном примере для некоторых комбинаций валюаций имеются инвариантные члены рядов более высоких степеней, чем для других.

Выпишем лорановы решения системы с полиномиальными коэффициентами, которая получается, если все $O(x^k)$ на рис. 1 равны 0. Эти решения вычислим до степени x^5 (для систем полного ранга с алгоритмически заданными рядами, частным случаем которых являются системы с полиномиальными коэффициентами, возможно вычислить усеченные решения до любой заданной степени). Решение приведено на рис. 3. Сравнивая рис. 2 и рис. 3, видим, что любое решение из множества на рис. 2 имеет свое продолжение на рис. 3. Это является следствием инвариантности всех членов рядов на рис. 2 относительно любых продолжений исходной системы, а полученная описанным способом система с полиномиальными коэффициентами является одним из возможных продолжений системы, представленной на рис. 1.

Множество решений системы с рис. 1 может быть задано и другим набором вектор-столбцов, например, приведенным на рис. 4. Можно заметить, что комбинации валюаций компонент векторов и соответствующие степени усечения на рис. 4 такие же, как и на рис. 2, и что на этих двух рисунках – два разных представления одного и того же множества усеченных лорановых решений с точностью до произвольных постоянных.

$$\begin{pmatrix} c_1x - c_1x^2 + \frac{3c_1}{4}x^3 - \frac{7c_1}{12}x^4 + \frac{47c_1}{96}x^5 + O(x^6), \\ c_2x^2 - \frac{4c_2}{3}x^3 + \frac{4c_2}{15}x^5 + O(x^6), \\ c_3 + \left(\frac{c_1}{2} + \frac{c_2}{2}\right)x^2 - \left(\frac{c_1}{3} + \frac{4c_2}{9}\right)x^3 + \frac{3c_1}{16}x^4 + \left(\frac{4c_1}{75} - \frac{7c_1}{60}\right)x^5 + O(x^6) \end{pmatrix}$$

Рис. 3.

$$\left[\begin{pmatrix} c_1x - c_1x^2 + O(x^3) \\ (-c_1 + 2c_3)x^2 + O(x^3) \\ c_2 + c_3x^2 + O(x^3) \end{pmatrix}, \begin{pmatrix} c_1x - c_1x^2 + O(x^3) \\ (-c_1 + 2c_3)x^2 + \left(\frac{4c_1}{3} - \frac{8c_3}{3}\right)x^3 + O(x^4) \\ c_3x^2 + O(x^3) \end{pmatrix}, \begin{pmatrix} c_1x - c_1x^2 + O(x^3) \\ -c_1x^2 + \frac{4c_1}{3}x^3 + O(x^4) \\ O(x^3) \end{pmatrix},\right.$$
$$\begin{pmatrix} 2c_3x - 2c_3x^2 + O(x^3) \\ O(x^3) \\ c_2 + c_3x^2 + O(x^3) \end{pmatrix}, \begin{pmatrix} 2c_3x - 2c_3x^2 + O(x^3) \\ O(x^4) \\ c_3x^2 + O(x^3) \end{pmatrix}, \begin{pmatrix} O(x^3) \\ 2c_3x^2 + O(x^3) \\ c_2 + c_3x^2 + O(x^3) \end{pmatrix},$$
$$\left.\begin{pmatrix} O(x^5) \\ 2c_3x^2 - \frac{8c_3}{3}x^3 + O(x^5) \\ c_3x^2 - \frac{8c_3}{9}x^3 + O(x^5) \end{pmatrix}, \begin{pmatrix} O(x^3) \\ O(x^3) \\ c_2 + O(x^3) \end{pmatrix}\right]$$

Рис. 4.

4. АЛГОРИТМ

4.1. Литералы

Как и в задаче поиска усеченных лорановых решений обыкновенного дифференциального уравнения с коэффициентами в виде усеченных рядов, алгоритм для систем основан на использовании *литералов* и построении решений, содержащих литералы. Литералами мы называем символьные обозначения незаданных коэффициентов рядов, входящих в системы (см. [5, 9]). Для ряда вида (2) будем говорить, что его коэффициенты при степенях x^s, $s > t_{kij}$, *не заданы*. Эти коэффициенты при построении решений усеченной системы мы и представляем символьными обозначениями – литералами.

Например, рассмотрим один из коэффициентов системы, представленной на рис. 1:

$$A_{0,3,1}(x) = -1 - x + \frac{1}{2}x^2 + O(x^3).$$

При x^0 коэффициент равен -1, при x^1 равен -1, при x^2 равен $\frac{1}{2}$. Незаданный коэффициент при x^3 обозначаем литералом $U_{[3,1][0,3]}$, при $x^4 - U_{[3,1][0,4]}$, и так далее. Здесь $U_{[i,j][k,s]}$ – выбранная в нашей реализации алгоритма форма представления литерала, обозначающего незаданный коэффициент при x^s в элементе $A_{kij}(x)$ матричного коэффициента $A_k(x)$ системы (1).

Отметим, что присвоение значений из K литералам в членах со степенями $t_{kij}+1,\ldots,t_{kij}+d_{kij}$ в рядах $A_{kij}(x)$, где d_{kij} – некоторые положительные целые числа, задает одно из возможных продолжений исходной системы. Соответственно, если имеется продолжение исходной системы, то оно соответствует значениям части литералов исходной системы. Если продолжение системы не содержит литералов, т.е. все ряды заданы полностью, то такое продолжение соответствует значениям всех литералов исходной системы.

4.2. Системы, к которым применим предлагаемый алгоритм

В данной работе рассматривается применение алгоритма только к таким системам линейных

обыкновенных дифференциальных уравнений (1), индуцированные рекуррентные системы (5) которых имеют невырожденную ведущую матрицу $B_0(n)$ и ее определитель не содержит литералов. Напомним, что в этом случае $\det B_0(n) = 0$ может рассматриваться как определяющее уравнение исходной дифференциальной системы – множество целых корней этого алгебраического уравнения включает множество всех возможных валюаций лорановых решений системы (1) и позволяет найти нижнюю границу валюаций всех лорановых решений системы, которая равна минимальному целому корню $\det B_0(n)$. Отсутствие литералов в $\det B_0(n)$ гарантирует инвариантность относительно любых продолжений системы (1) этого множества возможных валюаций и этой нижней границы.

Например, для рассмотренной в разд. 3 системы (см. рис. 1) ведущая матрица

$$B_0(n) = \begin{pmatrix} -1+n & 0 & 0 \\ 0 & 2n-n^2 & 0 \\ -1+n & -1+2n-n^2 & n \end{pmatrix}$$

индуцированной системы невырождена, ее определитель равен $(-1+n)n^2(2-n)$ и не содержит литералов.

Отметим, что матрица $B_0(n)$ может оказаться вырожденной и в том случае, когда ведущая матрица $A_r(x)$ системы (1) невырождена. В [1] показано, что для систем полного ранга с коэффициентами в виде рядов, заданных алгоритмически, эта ситуация не является тупиковой: алгоритм $\mathrm{EG}_\sigma^\infty$ позволяет преобразовать индуцированную систему $R(u) = 0$ к *охватывающей* ее рекуррентной системе с невырожденной ведущей матрицей и построить любое заданное число ее начальных слагаемых. Охватывающая рекуррентная система, дополненная множеством линейный ограничений, которые также строит алгоритм $\mathrm{EG}_\sigma^\infty$, имеет то же множество решений, что и система $R(u) = 0$. Это позволяет найти с любой заданной степенью усечения все лорановы решения $y(x)$ исходной дифференциальной системы (1) полного ранга. Здесь существенно, что множество линейных ограничений конечно и каждое из этих ограничений содержит лишь конечное число ненулевых слагаемых (благодаря тому, что нас интересуют решения $u(n)$, для которых $u_i(n) = 0$ при всех n, меньших нижней границы валюаций лорановых решений исходной дифференциальной системы). При использовании алгоритма $\mathrm{EG}_\sigma^\infty$ в случае усеченных систем возникает ряд дополнительных задач. Мы планируем решить их и представить результаты в дальнейших работах, расширив применимость алгоритма для случая, когда ведущая матрица индуцированной системы является вырожденной.

4.3. Построение усеченных лорановых решений

Новый алгоритм построения усеченных лорановых решений систем обыкновенных дифференциальных уравнений с коэффициентами в виде усеченных рядов является модификацией алгоритма построения усеченных лорановых решений систем обыкновенных дифференциальных уравнений с коэффициентами в виде алгоритмически заданных рядов. Для использования этого алгоритма коэффициенты системы в виде усеченных рядов приобретают алгоритмическое представление: если $s \le t_{kij}$, то применение алгоритма даст тот коэффициент, с которым x^s входит в усеченный ряд, а при $s > t_{kij}$ – литерал $U_{[i,j],[k,s]}$.

Подробное описание алгоритма построения лорановых решений систем с коэффициентами в виде алгоритмически заданных рядов дано в [1]. Этот алгоритм основывается на рассмотрении индуцированной системы (5). Множество целых корней определителя ведущей матрицы $B_0(n)$ содержит все возможные валюации лорановых решений (3). Проводя вычисления с помощью рекуррентной системы (5), последовательно определяем коэффициенты $u_1(n),\ldots,u_m(n)$ рядов в компонентах лорановых решений. Также вычисляются $B_{-s}(n)$ в (5) со все большими значениями s, в количестве, необходимом для вычисления требуемой степени усечения лорановых решений.

Вычисление последовательности $u(n)$ коэффициентов лоранова ряда (3), удовлетворяющей (5), выполняется последовательно, увеличивая значение n_0, начиная с $n_0 = \nu$ – минимального целого корня $\det B_0(n)$. Поскольку мы строим решения системы (5) такие, что $u(\nu-1) = 0$, $u(\nu-2) = 0$, ..., то для каждого целого n система (5) имеет конечное число ненулевых слагаемых в левой части. Для $n_0 = \nu$ получаем систему алгебраических уравнений неполного ранга

$$B_0(\nu)u(\nu) = 0$$

относительно неизвестных $u_1(\nu),\ldots,u_m(\nu)$. В решении этой системы некоторые компоненты вектора $u(\nu)$, возможно, будут *вычислены*, т.е. линейно выражены через другие компоненты, которые мы назовем на этом этапе *неизвестными постоянными*.

Если далее для некоторого целого $n_0 > \nu$ выполнено $\det B_0(n_0) \ne 0$, то

$$\begin{aligned} u(n_0) = -\big(B_0(n_0)\big)^{-1}\big(B_{-1}(n_0)u(n_0-1) + \\ + B_{-2}(n_0)u(n_0-2) + \ldots + B_{-n_0+\nu}(n_0)u(\nu)\big) \end{aligned} \qquad (6)$$

позволяет вычислить $u_1(n_0),\ldots,u_m(n_0)$ по ранее вычисленным $u_i(n_0-1),u_i(n_0-2),\ldots,u_i(v)$ $(i=1,...,m)$, т.е. получить линейное выражение $u_1(n_0),\ldots,u_m(n_0)$ через ранее введенные неизвестные постоянные.

Если же $\det B_0(n_0)=0$, то возникает система линейных алгебраических уравнений неполного ранга

$$B_0(n_0)u(n_0) = -B_{-1}(n_0)u(n_0-1) - \\ - B_{-2}(n_0)u(n_0-2) - \ldots - B_{-n_0+v}(n_0)u(v). \quad (7)$$

Вычислим ее решение $u_1(n_0),\ldots,u_m(n_0)$ с помощью метода Гаусса. Это приведет к тому, что некоторые компоненты $u_i(n_0)$ будут добавлены к множеству неизвестных постоянных. Также могут возникнуть соотношения, в которые не входят $u_i(n_0)$, но входят ранее введенные неизвестные постоянные. Эти соотношения позволят вычислить значения некоторых введенных ранее неизвестных постоянных. После того, как значение n_0 превзойдет наибольший целый корень $\det B_0(n)$, новые неизвестные постоянные возникать не будут и все дальнейшие $u(n_0)$ будут вычисляться по ранее вычисленным $u(n_0-1),u(n_0-2),\ldots,u(v)$. Неизвестные постоянные, оставшиеся после того, как n_0 превзойдет максимальный целый корень $\det B_0(n)$, объявляются *произвольными постоянными*, входящими в лораново решение дифференциальной системы (1). Произвольные постоянные обозначим через $c_1,\ldots,c_\tau$, где $\tau \le mr$.

При работе с усеченными системами используются литералы для представления незаданных коэффициентов. В алгоритм вносятся следующие дополнения, схожие с теми, которые использовались в алгоритме поиска усеченных лорановых решений обыкновенных дифференциальных уравнений с коэффициентами в виде усеченных рядов ([4, 5]). Согласно преобразованию (4), элементы $B_{-s,i,j}(n)$ матричных коэффициентов $B_{-s}(n)$ индуцированной системы (5) определяются следующим образом

$$B_{-s,i,j}(n) = \sum_{k=0}^{r} \Xi_{kij}(s)(n-s)^k, \quad (8)$$

где $s=0,1,\ldots$; $i=1,\ldots,m$; $j=1,\ldots,m$; Ξ_{kij} – алгоритм, возвращающий для целого числа s коэффициент при x^s ряда $A_{kij}(x)$ (см. (2)), если $s \le t_{kij}$, а в противном случае – литерал $U_{[i,j][k,s]}$. Таким образом, $B_{-s,i,j}(n)$ является для $s > \min_{0\le k\le r} t_{kij}$ полиномом от литералов над полем K. Напомним, что мы рассматриваем сейчас такие системы, что $\det B_0(n)$ не зависит от литералов, поэтому при вычислении $u_i(n_0)$ по (6) мы получим, что $u_i(n_0)$ – линейная комбинация неизвестных постоянных с полиномиальными от литералов коэффициентами.

В алгоритм вносим следующие модификации:

1. В ходе решения системы (7) могут возникнуть соотношения, содержащие литералы. В такие соотношения неизвестные постоянные входят линейно с коэффициентами, которые являются полиномами от литералов. Для инвариантности результата необходимо, чтобы при вычислении по соотношению какой-то неизвестной постоянной не происходило деление на зависящий от литералов полином. Если этого невозможно избежать (в соотношении все неизвестные постоянные имеют коэффициенты, содержащие литералы), то все неизвестные постоянные, входящие в это соотношение, получают значение ноль, поскольку в этом случае это значение является единственным инвариантным относительно всех возможных значений литералов, которые определяют все возможные продолжения исходной системы (1).

2. Вычисления с помощью индуцированной системы продолжаются до максимального целого корня $\det B_0(n)$ и далее, пока хотя бы одна компонента $y_i(x)$ решения $y(x)$ имеет коэффициент $u_i(n_0)$, не содержащий литералы, с учетом возможности приравнивания нулю различных комбинаций произвольных постоянных $c_1,\ldots,c_\tau$, не приводящих к обращению в ноль всех $u(v),\ldots,u(n_0-1)$.

3. После того, как вычисления завершены, происходит формирование набора $\bar{W}$ вектор-столбцов, представляющего множество W усеченных лорановых решений. Для этого приравниваются нулю произвольные постоянные $c_1,\ldots,c_\tau$ всеми возможными вариантами, не приводящими к обращению в ноль всех вычисленных $u(n)$. Это приводит к различным сочетаниям валюаций компонент решения $y(x)$. Для каждого такого сочетания валюаций в каждой компоненте $y_i(x)$ берутся коэффициенты $u_i(v)$, $u_i(v+1)$, … вплоть до $u_i(q_i)$ такого, что $u_i(q_i+1)$ – первый коэффициент, содержащий литералы.

4.4. Корректность алгоритма

Предложение 1. *Представленный алгоритм заканчивает свою работу.*

Доказательство: Рассмотрим целое число n_0, превышающее $r_{\max}$ – максимальный целый корень $\det B_0(n)$. Это значит, что $u(n_0)$ вычисляется по формуле (6) и каждая его компонента является линейной комбинацией произвольных постоянных $c_1,\ldots,c_\tau$ с коэффициентами-полиномами (возможно нулевой степени) от литералов. Пусть n_0 таково, что $n_0 - r_{max} > d$, где d – максимальная степень усечения среди всех элементов матричных коэф-

фициентов системы (1). Предположим, при таком n_0 для некоторой компоненты $u_i(n_0)$ хотя бы для одной произвольной постоянной c_k ее коэффициент b не зависит от литералов. В построенных алгоритмом $u(n_0-1), u(n_0-2), \ldots, u(v)$ положим нулю все c_j, кроме c_k. Полученные таким образом $\tilde{u}(n_0-1), \tilde{u}(n_0-2), \ldots, \tilde{u}(v)$, удовлетворяют индуцированной системе (5) для $n = v_k$, $v_k+1, \ldots, n_0-1$, где v_k – валюация усеченного лоранова решения $\tilde{y}(x)$ с коэффициентами $\tilde{u}(n)$, полученными таким вариантом приравнивания нулю части произвольных постоянных ($v \le v_k \le r_{\max}$, $\tilde{u}(v_k) \ne 0$, $\tilde{u}(v_k-1) = 0$, $\tilde{u}(v_k-2) = 0, \ldots, \tilde{u}(v) = 0$). Тогда

$$\tilde{u}(n_0) = -(B_0(n_0))^{-1}(B_{-1}(n_0)\tilde{u}(n_0-1) + B_{-2}(n_0)\tilde{u}(n_0-2) + \ldots + B_{-n_0+v_k}(n_0)\tilde{u}(v_k)). \quad (9)$$

Индуцированная система (5) строится с помощью преобразования (4). С учетом того, что все элементы матричных коэффициентов системы (1) являются усеченными рядами со степенью усечения не превышающей d, и того, что $n_0 - v_k \ge n_0 - r_{\max} > d$, получаем, что все элементы $B_{-n_0+v_k}(n_0)$ зависят от литералов, которые обозначают незаданные коэффициенты членов степени $n_0 - v_k$ рядов в системе (1), более того, слагаемые $B_{-1}(n_0)\tilde{u}(n_0-1)$, $B_{-2}(n_0)\tilde{u}(n_0-2)$, ..., $B_{-n_0+v_k-1}(n_0)\tilde{u}(v_k-1)$ именно этих литералов не содержат (см. (8)). Поскольку усеченное решение $\tilde{y}(x)$ имеет валюацию v_k, то хотя бы одна из компонент $\tilde{u}(v_k)$ – ненулевая. Тогда из (9) следует, что все компоненты $\tilde{u}(n_0)$ содержат литералы. В том числе $\tilde{u}_i(n_0) = bc_k$ содержит литералы, что противоречит предположению. □

Теперь покажем, что результат работы алгоритма соответствует постановке задачи.

Предложение 2. *Представленный алгоритм строит усеченные лорановы решения максимально возможной степени.*

Доказательство: В [1] показано, что алгоритм построения усеченных лорановых решений систем обыкновенных дифференциальных уравнений с коэффициентами в виде алгоритмически заданных рядов находит усечения всех лорановых решений заданной системы с такими коэффициентами. Новый алгоритм использует в своей работе этот же алгоритм, переходя от системы с коэффициентами в виде усеченных рядов к системе с алгоритмически заданными коэффициентами, где алгоритмы вместо незаданных коэффициентов возвращают соответствующие им литералы. Внесенные модификации исключают часть решений, которые не являются инвариантными по построению (см. модификация 1), продолжение вычислений таких решений зависит от того, равен ли нулю коэффициент при неизвестной постоянной. Таким образом оставшиеся решения являются инвариантными усеченными лорановыми решениями. Максимальность степени усечений обеспечена тем (см. модификации 2 и 3), что по построению в каждом вектор-столбце в каждой компоненте вычисленный коэффициент, имеющий степень, превышающую степень усечения, является полиномом, содержащим литералы. Это означает, что существуют разные продолжения исходной системы, соответствующие разным значениям литералов, такие, что значения этого полинома будут также различны, т.е. этот коэффициент не инвариантен. □

Представленный алгоритм имеет некоторую вариативность в своей работе. При вычислении с помощью индуцированной системы в возникающей системе (7) выбор тех неизвестных, которые будут вычислены, произволен. Различный выбор приводит к различным вариантам вычислений коэффициентов решения. Возникает вопрос, возможно ли, что различные варианты вычислений приведут к построению существенно отличающихся усеченных лорановых решений – например, один вариант даст некоторое решение с большей степенью усечения в одной из компонент, чем степень усечения в этой компоненте в любом из решений, найденных при ином варианте вычисления (в какой-то компоненте, возможно, будет обратная ситуация – степень усечения любого решения, построенного первым вариантом вычислений, будет меньше, чем степень усечения компоненты некоторого решения, построенного вторым вариантом).

Предложение 3. *Пусть $\bar{W}_1$ и $\bar{W}_2$ – два набора усеченных лорановых решений одной и той же системы вида (1), найденные представленным алгоритмом с двумя различными вариантами вычислений. Тогда $\bar{W}_1$ и $\bar{W}_2$ определяют одно и то же множество W усеченных лорановых решений системы (1).*

Доказательство: Рассмотрим систему S_p с полиномиальными коэффициентами, полученную из системы S путем принятия равным 0 всех коэффициентов рядов при степенях выше степени усечения ряда в коэффициентах системы. Такая система является частным случаем системы с алгоритмически заданными коэффициентами. Соответственно, с помощью алгоритма поиска усеченных лорановых решений систем с алгоритмически заданными коэффициентами возможно построить усеченные лорановы решения системы S_p со степенью усечения равной максимуму степеней усечения в компонентах вектор-столбцов в $\bar{W}_1$ и $\bar{W}_2$. Обозначим соответствующее множество усеченных решений S_p как W_p. Система S_p по построению является продолжением системы S. Поскольку, согласно предложению 2, как множе-

ство W_1, так и множество W_2 содержат усеченные лорановы решения, то любое усеченное решение как из W_1, так и из W_2 имеет продолжение в W_p и оба множества по построению могут рассматриваться как результат исключения из W_p тех решений, которые не являются инвариантными относительно продолжений системы S, т.е. их вычисление определяется конкретными значениями литералов системы S, которые они получили в системе S_p. Докажем утверждение предложения от противного. Пусть некоторое усеченное решение s имеется в W_1, но отсутствует в W_2. Это означает, что при вычислении по варианту, результатом которого является W_2, решение s было исключено из W_p. Это могло произойти только, если согласно этому варианту вычислений, было определено, что s неинвариантно зависит от литералов. Но это решение содержится в W_1, и поэтому от литералов не зависит. Противоречие. □

Наборы $\bar{W}_1$ и $\bar{W}_2$ определяют одно и то же множество усеченных решений W, но задающие их вектор-столбцы могут быть различны с точностью до произвольных постоянных. Наборы комбинаций валюаций компонент в вектор-столбцах, а значит и число вектор-столбцов, в $\bar{W}_1$ и $\bar{W}_2$ совпадают. Также совпадают степени усечений в одинаковых компонентах вектор-столбцов с одинаковыми комбинациями валюаций компонент в $\bar{W}_1$ и $\bar{W}_2$. Примером этого являются наборы, представленные на рис. 2, 4, – решения рассмотренной в разд. 3 системы.

Примечание 1. Если определитель $p(n)$ ведущей матрицы $B_0(n)$ индуцированной системы содержит литералы, можно показать, что существуют такие значения этих литералов, что ведущая матрица становится вырожденной, и что существуют значения литералов, при которых $p(n)$ имеет любой наперед заданный корень r_0 в дополнение к целым корням $r_1 \ldots r_k$ определителя $p(n)$, имеющимся вне зависимости от значения литералов. Таким образом, набор целых корней $p(n)$ не является инвариантным. Случай когда $\det B_0(n)$ содержит литералы пока исключен из случаев, охватываемых предложенным алгоритмом – задача определения возможности или невозможности вычисления усеченных лорановых решений в этом случае отложена для последующего рассмотрения.

5. РЕАЛИЗАЦИЯ И ПРИМЕРЫ

Алгоритм реализован в системе компьютерной алгебры `Maple 2022` ([10]) в виде модифицированной версии процедуры `LaurentSolution` пакета `TruncatedSeries` (описание текущей версии пакета приведено в работах [9–13], пакет и сессия Maple с примерами использования описываемых процедур доступны по адресу http://www.ccas.ru/ca/TruncatedSeries). Исходная версия процедуры `LaurentSolution` предназначена для случая линейных обыкновенных дифференциальных уравнений с коэффициентами в виде усеченных рядов. Модифицированная версия работает также и с системами таких уравнений. Эта версия частично основана на реализации процедур поиска лорановых решений для систем с алгоритмически заданными коэффициентами (см. [14]).

5.1. Аргументы и результат работы процедуры

Основные аргументы процедуры:

• Первый аргумент – система уравнений вида (1). Применение θ^k к вектор-столбцу неизвестных $y(x)$ записывается как `theta(y(x),x,k)`. Матричные коэффициенты системы записываются с помощью стандартной структуры `Matrix` в среде `Maple`. Элементы матриц задаются в виде выражений $a(x) + O(x^{t+1})$, где $a(x)$ – полином степени не выше t над полем алгебраических чисел, то есть аналогично математической записи. Умножение матрицы на столбец задается с помощью точки, как это принято в среде `Maple`.

• Второй аргумент – имя, обозначающее вектор-столбец неизвестных, например, `y(x)`.

Результат работы процедуры `LaurentSolution` при ее применении к системе уравнений – множество усеченных лорановых решений, представленное в виде списка вектор-столбцов из набора $\bar{W}$, описанного в разд. 3. Каждый вектор-столбец представлен списком компонент. Каждый элемент этого списка имеет вид

$$C_v x^v + C_{v+1} x^{v+1} + \ldots + C_q x^q + O(x^{q+1}),$$

где v – валюация данной компоненты, q – степень усечения, C_n – вычисленные коэффициенты лоранова ряда, которые являются линейными комбинациями произвольных постоянных вида $\_c_1$, $\_c_2$,… Если в качестве результата выдается пустое множество, значит лорановых решений не существует ни при каком продолжении заданной системы (определитель ведущей матрицы индуцированной системы не имеет целых корней). Также может быть возвращена константа `FAIL`. Это значит, что алгоритм не применим к заданной системе: ведущая матрица ее индуцированной системы вырожденная или ее определитель содержит литералы.

5.2. Примеры построения лорановых решений

Для использования модифицированной процедуры `LaurentSolution`, система, представ-

ленная на рис. 1, записывается в среде Maple следующим образом:

```
> s1 := Matrix(3,3,[[O(x^3),O(x^3),O(x^3)],
  [O(x^3),-1+x+x^2+O(x^3),O(x^3)],
  [O(x^3),-1+x+x^2+O(x^3),
  O(x^3)]]).theta(y(x),x,2)
  + Matrix(3,3,[[1+x+O(x^2),O(x^3),O(x^3)],
  [O(x^3),2-4*x-4*x^2+O(x^3),O(x^3)],
  [1+x+O(x^2),2-4*x-4*x^2+O(x^3),
  1 + O(x^5)]]).theta(y(x),x,1)
  +Matrix(3,3,[[-1+1/2*x^2+O(x^3),
  O(x^3),O(x^3)],[O(x^3),O(x^6),
  O(x^3)],[-1-x+1/2*x^2+O(x^3),
  -1+O(x^4), O(x^3)]]).y(x):
```

Применим процедуру к этой системе:

```
> LaurentSolution(s1, y(x));
```

$$\left[\left[-x^2\_c_1 + x\_c_1 + O(x^3), x^2\_c_2 + O(x^3),\ \_c_3 + x^2\left(\frac{\_c_1}{2} + \frac{\_c_2}{2}\right) + O(x^3)\right],\right.$$

$$\left[-x^2\_c_1 + x\_c_1 + O(x^3), x^2\_c_2 - \frac{4x^3\_c_2}{3} + O(x^4),\ x^2\left(\frac{\_c_1}{2} + \frac{\_c_2}{2}\right) + O(x^3)\right],$$

$$\left[x^2\_c_2 - x\_c_2 + O(x^3), x^2\_c_2 - \frac{4x^3\_c_2}{3} + O(x^4), O(x^3)\right],$$

$$\left[-x^2\_c_1 + x\_c_1 + O(x^3), O(x^3), \_c_3 + \frac{x^2\_c_1}{2} + O(x^3)\right],$$

$$\left[-x^2\_c_1 + x\_c_1 + O(x^3), O(x^4), \frac{x^2\_c_1}{2} + O(x^3)\right],$$

$$\left[O(x^3), x^2\_c_2 + O(x^3), \_c_3 + \frac{x^2\_c_2}{2} + O(x^3)\right],$$

$$\left[O(x^5), x^2\_c_2 - \frac{4x^3\_c_2}{3} + O(x^5),\ \frac{x^2\_c_2}{2} - \frac{4x^3\_c_2}{9} + O(x^5)\right], [O(x^3), O(x^3), \_c_3 + O(x^3)]\Bigg]$$

Результат вызова процедуры соответствует набору на рис. 2.

Рассмотрим следующую систему ([8], Ex.8):

$$\begin{pmatrix} 1+O(x^5) & O(x^5) \\ O(x^5) & 1-x+O(x^5) \end{pmatrix}\theta y(x) + \begin{pmatrix} O(x^5) & -1+O(x^5) \\ -x+2x^2+2x^3+2x^4+O(x^5) & -2+4x+O(x^5) \end{pmatrix} y(x) = 0$$

Эта система записывается в среде Maple следующим образом:

```
> s2 := Matrix(2,2,[[1+O(x^5),O(x^5)],
  [O(x^5),1-x+O(x^5)]]).theta(y(x),x,1)
  + Matrix(2,2,[[O(x^5),-1+O(x^5)],
  [-x+2*x^2+2*x^3+2*x^4+O(x^5),
  -2+4*x+O(x^5)]]).y(x):
```

Применим процедуру к этой системе:

```
> LaurentSolution(s2, y(x));
```

$$\left[\left[-x^3\_c_2 + x^2\_c_2 - x\_c_1 + \_c_1 + O(x^5),\ -3x^3\_c_2 + 2x^2\_c_2 - x\_c_1 + O(x^5)\right],\right.$$

$$\left[-x^3\_c_2 + x^2\_c_2 + O(x^7), -3x^3\_c_2 + 2x^2\_c_2 + O(x^7)\right]\Bigg]$$

Алгоритм из работы [8] находит, что эта система имеет следующие усеченные лорановы решения

$$\begin{pmatrix} C_1 - C_1x + C_2x^2 - C_2x^3 + O(x^4) \\ -C_1x + 2C_2x^2 - 3C_2x^3 + O(x^4) \end{pmatrix}$$

Видим, что новый алгоритм подтверждает этот результат и улучшает его – во-первых, в первом векторе из построенного множества степень усечения равна 4, а не 3, т.е. подтверждено, что в обеих компонентах решения нулевой коэффициент при степени x^4 будет при любом продолжении исходной системы, и, во-вторых, также дополнительно построены усечения до степени 6 для случая, когда валюации компонент решения равны 2.

Рассмотрим еще одну систему:

$$\begin{pmatrix} x+O(x^3) & O(x^3) \\ 1+O(x^3) & x+O(x^3) \end{pmatrix}\theta y(x) + \begin{pmatrix} O(x^3) & O(x^3) \\ O(x^3) & 3x+O(x^3) \end{pmatrix} y(x) = 0$$

Эта система записывается в среде Maple следующим образом:

```
> s3 := Matrix(2,2,[[x+O(x^3),O(x^3)],
  [1+O(x^3),x+O(x^3)]]).theta(y(x),x,1)
  +Matrix(2,2,[[O(x^3),O(x^3)],
  [O(x^3),3*x+O(x^3)]]).y(x):
```

Применим процедуру к этой системе:

```
> LaurentSolution(s3, y(x));
```

FAIL

Такой результат означает, что предложенный алгоритм не может найти усеченные лорановы решения этой системы. В данном случае, ведущая матрица индуцированной рекуррентной системы

$$B_0(n) = \begin{pmatrix} 0 & 0 \\ n & 0 \end{pmatrix}$$

является вырожденной, соответственно, предложенный алгоритм не применим к этой системе. В дальнейших работах мы предполагаем доработать алгоритм и обеспечить его применимость в том числе и к этой системе.

6. ЗАКЛЮЧЕНИЕ

В настоящей работе предложен алгоритм построения усеченных лорановых решений системы линейных обыкновенных дифференциальных уравнений с усеченными степенными рядами в роли коэффициентов для случая, когда ведущая матрица в индуцированной рекуррентной системе этой дифференциальной системы невырождена и имеет определитель, не содержащий литералы. Очевидно, что не все системы обладают таким свойством, поэтому в дальнейшем мы планируем расширить применимость алгоритма на случай вырожденной ведущей матрицы индуцированной рекуррентной системы. Мы планируем использовать версию EG_σ^∞ алгоритма EG ([15]), предназначенную для систем с коэффициентами в виде алгоритмически заданных рядов. EG_σ^∞ преобразует индуцированную рекуррентную систему к охватывающей ее системе с невырожденной ведущей матрицей. Предстоит решить задачу учета наличия литералов в ходе работы алгоритма EG_σ^∞. Также мы планируем исследовать случай, когда определитель ведущей матрицы содержит литералы.

СПИСОК ЛИТЕРАТУРЫ

1. *Abramov S.A., Barkatou M.A., Khmelnov D.E.* On full rank differential systems with power series coefficients // J. of Symbolic Computation. 2015. V. 68. P. 120–137.
2. *Абрамов С.А., Хмельнов Д.Е.* Регулярные решения линейных дифференциальных систем с коэффициентами в виде степенных рядов // Программирование. 2014. № 2. С. 75–85.
3. *Рябенко А.А.* Экспоненциально-логарифмические решения линейных дифференциальных систем с коэффициентами в виде степенных рядов // Программирование. 2015. № 2. С. 54–62.
4. *Абрамов С.А., Рябенко А.А., Хмельнов Д.Е.* Линейные обыкновенные дифференциальные уравнения и усеченные ряды // Ж. выч. мат. и мат. физ. 2019. Т. 59. № 10. С. 66–77.
5. *Абрамов С.А., Рябенко А.А., Хмельнов Д.Е.* Регулярные решения линейных обыкновенных дифференциальных уравнений и усеченные ряды // Ж. выч. мат. и мат. физ. 2020. Т. 60. № 1. С. 4–17.
6. *Абрамов С.А., Рябенко А.А., Хмельнов Д.Е.* Усеченные ряды и формальные экспоненциально-логарифмические решения линейных обыкновенных дифференциальных уравнений// Ж. вычисл. матем. и матем. физ. 2020. Т. 60. № 10. С. 1664–1675.
7. *Abramov S.A., Barkatou M.A., Pfluegel E.* Higher-order linear differential systems with truncated coefficients // In Proc. of CASC'2011, 2011. P. 10–24.
8. *Abramov S.A., Barkatou M.A.* On Strongly Non-Singular Polynomial Matrices // In: *Schneider C., Zima E.* (eds) Advances in Computer Algebra. Springer Proceedings in Mathematics & Statistics. 2018. V. 226. P. 1–17.
9. *Абрамов С.А., Рябенко А.А., Хмельнов Д.Е.* Процедуры поиска лорановых и регулярных решений линейных дифференциальных уравнений с усеченными степенными рядами в роли коэффициентов // Труды ИСП РАН. 2019. Т. 31. № 5. С. 233–248.
10. Maple online help // http://www.maplesoft.com/support/help/
11. *Abramov S., Khmelnov D., Ryabenko A.* Truncated and infinite power series in the role of coefficients of linear ordinary differential equations // Proc. CASC'2020. Lecture Notes in Computer Science. 2020. V. 12291. P. 63–76.
12. *Abramov S., Khmelnov D., Ryabenko A.* The Truncated-Series Package for Solving Linear Ordinary Differential Equations Having Truncated Series Coefficients// In: Maple in Mathematics Education and Research, Springer Nature Switzerland. 2021. P. 19–33.
13. *Абрамов С.А., Рябенко А.А., Хмельнов Д.Е.* Процедуры поиска усеченных решений линейных дифференциальных уравнений с бесконечными и усеченными степенными рядами в роли коэффициентов// Программирование. 2021. № 2. С. 56–65.
14. *Абрамов С.А., Рябенко А.А., Хмельнов Д.Е.* Процедуры поиска локальных решений линейных дифференциальных систем с бесконечными степенными рядами в роли коэффициентов // Программирование. 2016. № 2. С. 75–86.
15. *Abramov S.* EG-eliminations. J. of Difference Equations and Applications. 1999. V. 5. P. 393–433.

Searching for Laurent Solutions of Systems of Linear Differential Equations with Truncated Power Series in the Role of Coefficients

© 2023 г. S. A. Abramov[a,#], A. A. Ryabenko[a,##], and D. E. Khmelnov[a,###]

[a] *Federal Research Center "Computer Science and Control," Russian Academy of Scienc, Moscow, Russia*

[#] *e-mail: sergeyabramov@mail.ru*

[##] *e-mail: anna.ryabenko@gmail.com*

[###] *e-mail: dennis_khmelnov@mail.ru*

Systems of linear ordinary differential equations with the coefficients in the form of infinite formal power series are considered. The series are represented in a truncated form, with the truncation degree being different for different coefficients. Induced recurrent systems and literal designations for unspecified coefficients of the series are used as a tool for studying such systems. An algorithm for constructing Laurent solutions of the system is proposed for the case where the determinant of the leading matrix of the induced system is not zero and does not contain literals. The series included in the solutions are still truncated. The algorithm finds the maximum possible number of terms of the series that are invariant with respect to any prolongations of the truncated coefficients of the original system. The implementation of the algorithm as a Maple procedure and examples of its usage are presented.

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 47–58

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 004.421.6

О РЕАЛИЗАЦИИ ЧИСЛЕННЫХ МЕТОДОВ РЕШЕНИЯ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В СИСТЕМАХ КОМПЬЮТЕРНОЙ АЛГЕБРЫ

© 2023 г. А. Баддур[a], М. М. Гамбарян[a], Л. Гонсалес[a], М. Д. Малых[a,b,*]

[a] *Российский университет дружбы народов*
117198 Москва, улица Миклухо-Маклая, 6, Россия
[b] *Объединенный институт ядерных исследований*
141980 Дубна Московской области, Россия
**E-mail: malykh_md@pfur.ru*

Поступила в редакцию 21.06.2022 г.
После доработки 28.07.2022 г.
Принята к публикации 30.10.2022 г.

В статье представлен оригинальный пакет для исследования численных решений обыкновенных дифференциальных уравнений, встраиваемый в систему компьютерной алгебры Sage. Этот проект направлен на более тесную интеграцию численных и символьных методов и прежде всего преследует цель создания удобного инструмента для работы с численными решениями в Sage. В этом пакете определено два новых класса – начальные задачи и приближенные решения. Внутри первого класса определены инструменты для символьных вычислений, связанных с начальными задачами, внутри второго – инструменты для интерполяции значений символьных выражений на приближенном решении и оценивания ошибки по методу Ричардсона. Затем кратко описана реализация метода Рунге–Кутты, главная особенность которой – возможность работы с произвольными таблицы Бутчера и произвольными числовыми полями.

DOI: 10.31857/S013234742302005X, **EDN:** GZBPBR

1. ВВЕДЕНИЕ

Ввиду востребованности интегрирования дифференциальных уравнений при решении прикладных задач первые интеграторы были созданы на заре появления компьютерной техники [1]. Численные интеграторы обыкновенных дифференциальных уравнений (ОДУ), находящиеся сейчас во всеобщем употреблении, были разработаны в 1980-х годах. Ряд удачных разработок был относительно недавно переписан на python и доступен в библиотеке SciLab [2]. Среди многочисленных альтернатив этому собранию, следует выделить проект nodepy (`https://github.com/ketch/nodepy`), позволяющий проводить компьютерные эксперименты со схемами высокого порядка [3].

При сравнении разностных схем внимание исследователей всегда было сосредоточено на количественной близости точных и приближенных решений, а вопрос о сохранении качественных свойств точного решения до сих пор остается недостаточно изученным. Заманчивая возможность “определить характер динамического процесса с использованием только грубых вычислений с большим шагом сетки” по симплектической схеме была отмечена в [4]. В теории дифференциальных уравнений в частных производных дискретизации, наследующие некоторые свойства дифференциальных уравнений, называются миметическими (mimetic), то есть подражающими [5, 6]. Поэтому представляется целесообразным рассматривать разработку новых разностных схем в контексте концепции наследования алгебраических и качественных свойств динамических систем, разумеется, уточняя саму концепцию подражания.

Исторически первым был выделен класс схем Рунге–Кутты, сохраняющих симплектическую структуру гамильтоновых динамических систем. Из общих соображений можно было бы ожидать, что сохранение симплектической структуры должно иметь следствием сохранение всех алгебраических интегралов движения, однако оказалось, что сохраняются точно только линейные и квадратичные интегралы [7]. Можно конструировать разностные схемы, сохраняющие все алгебраические интегралы движения динамической системы, напр., задачи многих тел, однако при этом приходится жертвовать гамильтоновой структурой [8, 9], а можно – схемы, сохраняющие свойство обратимости исходной динамической системы [10].

Инструменты, необходимые для проектирования и исследования алгебраических свойств таких схем, уже имеются в системах компьютерной алгебры. Однако сами эти системы предоставляют весьма бедный инструментарий для работы с приближенными решениями, поскольку этот вопрос традиционно относится к численным методам. Напр., в Sage [11] по умолчанию доступен один единственный численный метод решения ОДУ – метод Рунге–Кутты 4-го порядка с постоянным шагом. При этом сама реализация, очевидно, не писалась специально для Sage, поскольку она не позволяет менять поле, над которым ведутся расчеты.

В настоящей статье представлен оригинальный пакет `fdm`, встраиваемый в систему компьютерной алгебры Sage. Наша цель – создать удобный инструмент для численно-аналитического исследования разностных схем, хорошо интегрированный с алгебраическим инструментарием.

2. ОПИСАНИЕ НАЧАЛЬНОЙ ЗАДАЧИ

В известных нам системах задание начальной задачи и отыскание ее приближенного решения выполняется в одно действие. Такой подход препятствует четкому разделению корректно поставленной математической задачи и метода ее решения. Поэтому в нашем пакете начальная задача описывается отдельно от метода ее решения как элемент специально созданного для этого класса `Initial_problem`. Начальная задача

$$\begin{cases} \dfrac{dx_i}{dt} = f_i(t, x_1, \ldots, x_n), \quad i = 1, \ldots, n \\ x_i(0) = x_i^{(0)}, \quad i = 1, \ldots, n \end{cases} \tag{2.1}$$

на отрезке $0 < t < T$ описывается списком $[x, f, x_0, T]$, где $x = [x_1, \ldots, x_n]$ – список используемых переменных, $f = [f_1, \ldots, f_n]$ – список правых частей, элементы которого являются символьными выражениями, $x_0 = [x_1^{(0)}, \ldots, x_n^{(0)}]$ – список начальных данных, и T – конечное время. Начальные задачи рассматриваются как элементы класса `Initial_problem`.

Напр., начальная задача

$$\begin{cases} \dfrac{dx_1}{dt} = x_2, \quad \dfrac{dx_2}{dt} = -x_1, \\ x_1(0) = 0, \quad x_2(0) = 1 \end{cases} \tag{2.2}$$

на отрезке $0 < t < 10$ описывается так:

```
var("t,x1,x2")
pr1=Initial_problem([x1, x2], [x2, -x1],
[0,1], 10)
```

В нашей системе независимая переменная всегда обозначается как t и интерпретируется как время, за начальный момент всегда принимается $t = 0$. Это соглашение позволяет использовать в аргументах `Initial_problem` списки одной длины. В стандартной реализации метода Рунге–Кутты в Sage эти списки отличаются, что является источником многочисленных опечаток при ее использовании.

Начальные условия $x_1^{(0)}, \ldots, x_n^{(0)}$ и конечный момент времени T являются "числами", однако с точки зрения компьютерной алгебры обычно они задаются символьными выражениями, которые можно преобразовать в элемент поля вещественных чисел $\mathbb{R}$. Напр., в качестве начального значения может быть использован и $\sqrt{2}$, и sin1. Преобразование в десятичные дроби такого рода выражений вносит ошибку округления и поэтому в нашей системе делается на стадии отыскания численного решения.

Функция `latex`, определенная в этом классе, возвращает начальную задачу в нотации LaTeX, что очень удобно для проверки корректности задания задачи.

Выделение начальных задач в особый класс позволило реализовать внутри этого класса вычисление в символьном виде ряда полезных для пользователя конструкций. Дело в том, что не зная решения начальной задачи, мы можем вычислить в символьном виде полную производную по t любого символьного выражения u, зависящего явно от x, t, многочлен Тейлора для этой функции, кривизну интегральной кривой в заданной точке и т.п. Приведем несколько примеров.

Для начальной задачи (2.2) полная производная функции $u = x_1^2 + x_2^2$ вычисляется так:

```
pr1.diff(x1^2+x2^2)
```

В данном случае возвращается нуль. Разложение в ряд Тейлора функции $v = x_1^2$ до членов второго порядка вычисляется так:

```
pr1.taylor(x1^2,2)
```

$$-(x_1^2 - x_2^2)(t - \tau)^2 - 2(t - \tau)x_1x_2 + x_1^2$$

Здесь τ – центр разложения. Кривизна интегральной кривой в точке (x_1, x_2) вычисляется так:

```
pr1.curvature()
```

$$\frac{(x_1^2 + x_2^2)^2 + x_1^2 + x_2^2}{(x_1^2 + x_2^2 + 1)^{\frac{3}{2}}}$$

В будущем мы планируем интегрировать в этот класс инструменты для отыскания рациональных интегралов, используя программное обеспечение для Sage, созданное в рамках диссертационного исследования Юй Ин [12].

Для генерации начальной задачи тоже можно использовать символьные вычисления, что осо-

бенно удобно в случае, если тел действительно много.

3. ИНСТРУМЕНТЫ ДЛЯ РАБОТЫ С ЧИСЛЕННЫМИ РЕШЕНИЯМИ НАЧАЛЬНОЙ ЗАДАЧИ

Численное решение задачи (2.1) выполняется над некоторым полем. Условимся обозначать поле, над которым мы работаем, как $\mathbb{K}$. По умолчанию в качестве такого используется стандартная реализация поля вещественных чисел $\mathbb{R}$ как множества десятичных дробей с плавающей запятой и фиксированным количеством бит, используемых для представления мантиссы. Как уже отмечалось выше, пользователь не обязан задавать начальные условия как элементы этого класса, но система должна иметь возможность преобразовать начальные условия в элементы поля $\mathbb{K}$.

Численное решение задачи (2.1) представляет собой список точек, каждый элемент которого – список координат точек решения: первым идет значение t, а затем $x_1,\ldots,x_n$ в том порядке, в котором они идут в x. Здесь используется тот же формат списка, что и в стандартной реализации метода Рунге-Кутты в Sage. Однако сами числа, входящие в списки, не обязаны принадлежать именно полю $\mathbb{R}$, но должны принадлежать полю $\mathbb{K}$.

Целый ряд манипуляций с этим списком можно делать независимо от метода, которым он был получен, поэтому мы создали для него особый класс – `Numsol`. Помимо списка точек приближенного решения элемент этого класса содержит указание на начальную задачу, порядок аппроксимации этой задачи разностной схемой и шаг равномерной сетки Δt или аналог h этой величины для квазиодномерной сетки.

В нашем пакете все решатели возвращают элемент класса `Numsol`. Напр., явный метод Рунге–Кутты реализован в виде функции `erk`, аргументами которой служат начальная задача и, опционально, число N отрезков, на которые делится рассматриваемый интервал (см. ниже). Она возвращает не список точек приближенного решения, но элемент класса `Numsol`. Напр., для задачи (2.2)

```
Q=erk(pr1,N_)
```

Конечно, предусмотрена возможность вывести список точек приближенного решения:

```
Q.list()
```

Однако пользователь может и не знать об этой возможности, поскольку в нашем пакете предусмотрена возможность вывести значение любого символьного выражения u от t,x в любой точке отрезка $[0,T]$. Напр., значение выражения x_1+t при $t=\pi$ можно найти так:

```
sage: Q.value(x1+t, pi)
```

3.15802440240400.

Для вычисления значения выражения u в точках, не попавших в узлы разбиения отрезка $[0,T]$, используется аппроксимация многочленом Тейлора, порядок которого согласован с порядком аппроксимации, указанным решателем при создании решения как элемента класса `Numsol`.

Замечание. Мы пытались использовать вместо затратной операции разложения по формуле Тейлора сплайны. Однако встроенная в Sage функция для аппроксимации сплайнами не позволяет менять поле $\mathbb{K}$ и ее не удалось согласовать с порядком аппроксимации разностной схемы при больших порядках аппроксимации.

Метод `zeros` позволяет найти нули выражения u на отрезке $0 < t \le T$. Напр., найдем нули x_1:

```
Q.zeros(x1)
```

[3.1430180411731743,…]

Определение нулей происходит по перемене знака, поэтому нули четной кратности найдены не будут.

Метод `plot` позволяет построить двумерный график зависимости одного символьного выражения от любого другого. Напр., график зависимости $x_1^2 - x_2^2$ от $x_1^2 + x_2^2$ можно вывести так:

```
sage: Q.plot(x1^2+x2^2,x1^2-x2^2)
```

При этом оси подписываются автоматически, при $N < 51$ указываются точки приближенного решения, при большем числе точек они соединяются сплошной линией. Этот метод поддерживает стандартные опции: `axes_labels`, `linestyle`, `color`. Остальные опции, традиционно используемые в графике для Sage, можно определить через метод `show`. Таким образом, вычисление символьного выражения на решении и построение его графика не требуют от пользователя понимания того, как устроено численное решение.

Для того, чтобы охарактеризовать численный метод, полезно использовать метод `plot_dt`, который строит двумерный график зависимости шага от t:

```
Q.plot_dt()
```

В данном случае, шаг постоянный и зависимость тривиальная.

4. РЕАЛИЗАЦИЯ МЕТОДА РИЧАРДСОНА

В работах Ричардсона для оценок ошибок, возникающих при вычислении определенных интегралов по методу конечных разностей, было предложено сгущать сетку, а в работах Рунге схожий прием был приложен к исследованию обыкновенных дифференциальных уравнений. Этот подход был систематически разработан в работах

Н.Н. Калиткина и его учеников [13–16] и будет далее называться метод Ричардсона.

На наш взгляд, этот метод особенно просто описывается как метод оценки ошибки вычисления значения символьного выражения u в заданной точке $t = a$ [17]. Итак, пусть задана начальная задача, момент времени $t = a$, попадающий на интервал $0 < a \le T$, и символьное выражение u, зависящее от $x_1,\ldots,x_n$ и t. Найдем одним и тем же методом, но с разным шагом $h = h_1, h_2,\ldots$, приближенные решения этой начальной задачи (1). Затем вычислим значения $u_1, u_2,\ldots$ выражения u в точке $t = a$ для каждого из этих решений, используя при необходимости описанную выше интерполяцию. Если порядок аппроксимации равен r и интерполяция согласована с этим порядком, то

$$u_i - u = ch_i^r + c' h_i^{r+1} + \ldots, \quad i = 1,2,\ldots \qquad (4.1)$$

Здесь $c, c',\ldots$ – неизвестные нам константы, которые не зависят от шага и характеризуют начальную задачу и метод ее решения, но, конечно, зависят от рассматриваемого момента времени $t = a$. Если $c \ne 0$, то при достаточно малых h можно оставить только главный член, отсюда

$$u_i - u = ch_i^r.$$

В таком случае

$$u_1 - u_2 = c(h_1^r - h_2^r).$$

Поэтому ошибку аппроксимации

$$u \simeq u_2$$

можно оценить как

$$ch_2^r = \frac{u_1 - u_2}{\left(\frac{h_1}{h_2}\right)^r - 1}.$$

В нашем пакете функция `richardson(P,Q,u,a)` возвращает по двум численным решениям P, Q значение символьного выражения u при $t = a$ и оценку совершаемой при этом ошибки E. Формат ответа согласован с функций `numerical_integral`, используемой в Sage для численного вычисления интегралов.

Пример 1. Рассмотрим решение задачи

$$\frac{dx}{dt} = t^2 + x, \quad x(0) = 0 \qquad (4.2)$$

на отрезке $0 < t < 1$ по методу `rk4` (см. п. 5.2). Найдем, напр., значение выражения $u = x^2$ при $t = 0.8$ и оценим ошибку по методу Ричардсона:

```
var("x,t")
pr2=Initial_problem(x,t^2+x,0,1)
P=erk(t^2+x,x,0, T=1, N=10)
Q=erk(t^2+x,x,0, T=1, N=20)
richardson(P,Q,x^2,0.8)
```

$$[0.0445555409432967, -7.89293834874139\times10^{-9}]$$

Замечание. Традиционно, интересуются суммой $u_2 + E$, которую интерпретируют как уточнение значения u по методу Ричардсона. Однако, на наш взгляд два числа – u_2, E – важны порознь: второе характеризует порядок ошибки численного метода. Разумеется, пользователь может сложить два элемента списка, который возвращает функция `richardson` и получить уточнение по Ричардсону.

Замечание Следует подчеркнуть, что наша оценка носит локальный характер, она зависит от выбора $t = a$. Из общих соображений можно ожидать, что с ростом a ошибка становится все больше и больше. Однако мы сталкивались со случаями, когда зависимость ошибки от a не была монотонной, напр., при исследовании системы Калоджеро [18].

Метод Ричардсона дает неверные оценки, если следующие члены ряда (3) все еще велики. Так будет, напр., если в рассматриваемой точке $t = a$ имеется суперсходимость, то есть $c = 0$ и порядок аппроксимации выше чем порядок аппроксимации разностной схемы. Поэтому перед применением метода Ричардсона рекомендуется убедиться в том, что в рассматриваемом диапазоне изменения шага h зависимость ошибки E от h является степенной и оценить ее показатель r. Для этого следует вычислить десяток-другой решений

$$L = [P_1, P_2,\ldots]$$

с разным шагом и построить зависимость E от h в двойном логарифмическом масштабе (диаграмма Ричардсона). Если эта зависимость – степенная ($E = ch^r$), то должна получиться прямая, наклон которой равен r. Мы, конечно, не знаем ошибку точно, поэтому вместо них используются оценки по Ричардсону.

В нашем пакете функция

`richardson_plot(L,u,a)`

отмечает на плоскости h, E в двойной логарифмической шкале точки

$$(h_i, E_i),$$

где h_i – значение шага, а E_i – оценки ошибки в вычислении значения выражения u при $t = a$ по Ричардсону. К этой диаграмме добавляется прямая, которая проходит к этим точкам ближе всего в смысле метода наименьших квадратов. Метод Ричардсона применим в рассматриваемом диапазоне изменения шагов, если точки с графической точностью ложатся на прямую, а наклон прямой

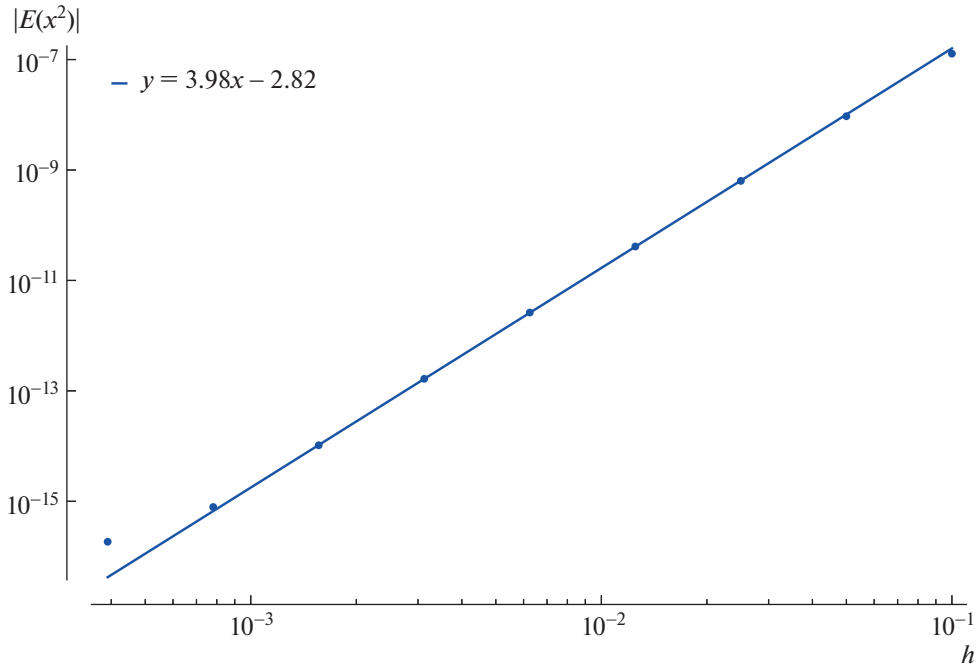

Рис. 1. Диаграмма Ричардсона для значения x_1^2 при $t = 0.8$ для примера 1.

указывает на фактический порядок аппроксимации.

Пример 2. Возвращаясь к примеру 1, мы можем построить диаграмму Ричардсона следующим образом:

```
@parallel
def foo(n):
    return erk(pr2, N=2^n*10)
L = list(foo(list(range(10))))
richardson_plot([L[i] [1] for i in
range(len(L))], x^2, 0.8, nmin=2, nmax=7)
```

Функция `richardson_plot(L,u,a)` имеет две опции (`nmin` и `nmax`), позволяющие исключить из аппроксимации прямой первые и последние точки. Результат представлен на рис. 1. Хорошо видно, что наклон равен 3.98 против ожидаемых 4. При больших шагах оказывают приметное влияние следующие члены в разложении (3), а при слишком малых h сказывается ошибка округления.

Поскольку расчеты численных решений при различных шагах никак друг с другом не связаны, построение диаграммы Ричардсона допускает естественное распараллеливание, использованное в примере 2. Благодаря декоратору `@parallel`, встроенному в Sage, на многоядерных процессорах построение диаграммы занимает столько же времени, сколько вычисление одного решения с самым мелким шагом.

Если этот порядок заметно отличается от порядка аппроксимации, то при дальнейшем применении функции `richardson` следует воспользоваться опцией `delta`, значение которой указывает на сколько следует увеличить порядок.

Метод Ричардсона может быть применен и к вычислению нулей. Напр., пусть требуется найти по методу Рунге–Кутты нули решения $x_1 = \sin t$ задачи (2.2), обозначенной выше как задача `pr1`. Вычислим список L численных решений, постепенно сгущая сетку:

```
L=[erk(pr1, N=2^n*10) for n in range(10)]
```

Вычислим нули по наиболее точному из решений:

```
L[-1].zeros(x1)
```

[3.141592653590175,…]

Функция `richardson_plot_zeros` позволяет построить диаграмму Ричардсона для каждого из нулей (нули считаются слева направо, начиная с нуля; указываются путем задания опции num):

```
richardson_plot_zeros(L, x1, num=2)
```

Результат представлен на рис. 1. Хорошо видно, что наклон графика ожидаемо равен 4, а ошибка на последних элементах списка L имеет порядок 10^{-12}. Более точно:

```
richardson_zeros(L[-1],L[-2], x1, num=2)
```

$[9.424777960770523, 1.1424342953129477 \times 10^{-12}]$

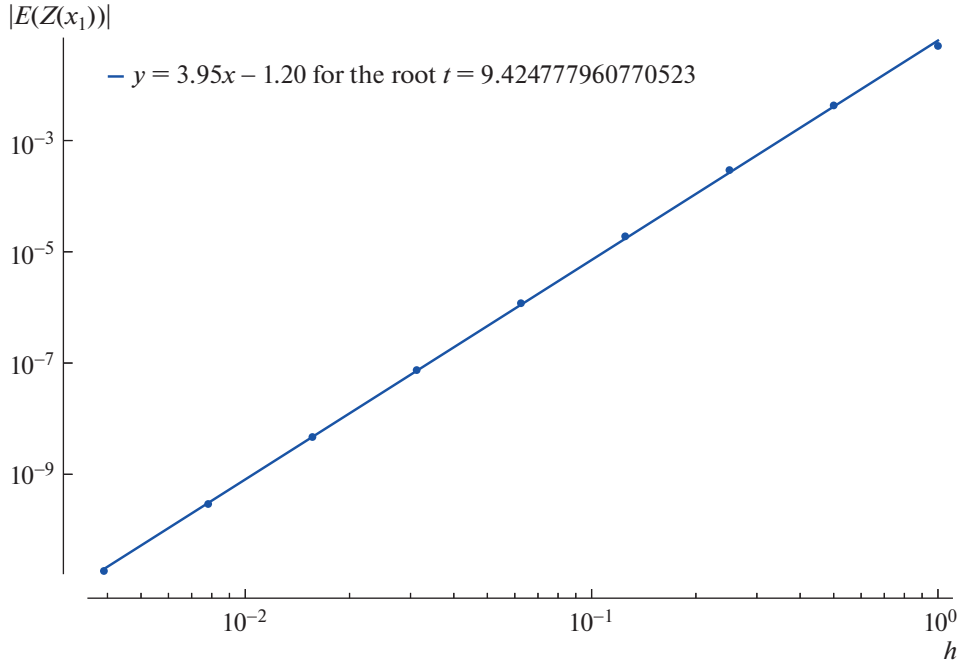

Рис. 2. Диаграмма Ричардсона для последнего нуля x_1.

Таким образом, 3π вычисляется с точностью до 11 знака после запятой, в чем в данном случае можно убедиться непосредственной проверкой.

5. РЕАЛИЗАЦИЯ МЕТОДА РУНГЕ–КУТТЫ

Для однозначной идентификации метода Рунге–Кутты достаточно указать таблицу Бутчера [19]. При реализации метода Рунге–Кутты в пакете fdm мы стремились к тому, чтобы эта реализация могла работать с любыми полями и любыми матрицами Бутчера. В настоящей момент доступны 2 реализации: отдельно для явного, отдельно для неявного методов. Первая написана Л. Гонсалесом, вторая – Али Баддуром.

5.1. Таблицы Бутчера

В нашем пакете для работы с таблицами Бутчера мы завели отдельный класс `Butcher_tableau`. Для задания таблицы нужно указать порядок аппроксимации и саму таблицу в виде списка, первым элементом которой будет матрица a, а вторым – столбец b. Столбец c вычисляется по этим данным. Опционально далее указываются краткое название метода, основанного на этой таблице, и описание. Напр., для задания таблицы Бутчера стандартного метода Рунге–Кутты

$$\begin{array}{c|cccc} \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & & \frac{1}{2} & & \\ 1 & & & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

мы пишем:

```
rk4=Butcher_tableau(4,[[[0,0,0,0],
[1/2,0,0,0], [0,1/2,0,0],[0,0,1,0]],
 [1/6,1/3,1/3,1/6]], 'rk4',
 'Standard rk method')
```

Функции, определенные в классе `Butcher_tableau`, носят вспомогательный характер и используются в нашей реализации метода Рунге–Кутты. Для пользователя может быть полезна лишь одна из них – `latex`, которая возвращает таблицу Бутчера в нотации LaTeX.

В наш пакет интегрирована функция `butcher_eqs(p,s)`, написанная Юй Ин [20], эта функция возвращает систему уравнений на элементы таблицы Бутчера по заданным порядку p и числу стадий s и опции implicit, которая может принимать два значения True/False.

Пример 3. Двустадийная явная таблица Бутчера имеет вид

$$\begin{array}{c|cc} c_0 & & \\ c_1 & a_{10} & \\ \hline & b_0 & b_1 \end{array}$$

Эта таблица задает разностную схему порядка $p = 2$, если ее коэффициенты удовлетворяют двум уравнениям, которые возвращает функция `butcher_eqs`:

```
butcher_eqs(2,2, implicit=False)
```

$$-b_0 - b_1 + 1 = 0, \quad -2a_{10}b_1 + 1 = 0. \qquad (5.1)$$

Поэтому имеется однопараметрическое семейство явных методов Рунге–Кутты второго порядка с двумя стадиями.

В 2000-х годах появились первые программы для численного, а позже и символьного решения систем уравнений для определения элементов таблицы Бутчера [21–25]. Большая коллекция таблиц Бутчера до 12 порядка аппроксимации была собрана П. Стоуном [26], который производил вычисления в системе Maple. Вычислять с нуля каждый раз таблицы Бутчера большого порядка весьма неразумно, поэтому в fdm имеется коллекция таблиц Бутчера, реализованная пока в виде списка `butchers_list`, который задан в файле `butchers_list.sage`. Эта коллекция в существенном основана на коллекции Стоуна.

Замечание. При перенесении таблиц Бутчера высокого порядка возникают неизбежные опечатки, таблицы, используемые в нашей системе, проверены по методу Ричардсона на тестовых задачах. На данный момент, одна из таблиц, взятых на сайте Стоуна, отмечена в `butchers_list.sage` как сомнительная.

5.2. Явный метод Рунге–Кутты

Функция `erk` реализует явный метод Рунге–Кутты, при этом пользователь может выбрать любую матрицу Бутчера из базы или указать свою. По умолчанию используется матрица rk4. Аргументами этой функции служит начальная задача. Предусмотрено еще две опции: N – число отрезков, на которое делится производный отрезок $[0, T]$ и `tableau` – таблица Бутчера.

По методу Ричардсона мы можем проверить правильность определения порядка аппроксимации, что весьма существенно для корректного заполнения базы таблиц Бутчера.

Пример 4. В качестве первого примера рассмотрим начальную задачу (4) на отрезке $0 < t < 1$, точное решение которой известно:

```
x_exact=-t^2 - 2*t + 2*e^t - 2
```

Сравним его с двумя решениями, найденными при помощи двух различных матриц Бутчера. В качестве первой таблицы возьмем стандартную таблицу метода rk4.

```
P=erk(pr2, N=10)
Q=erk(pr2, N=20)
richardson(P,Q,x,0.8)
```

$$[0.211081834707056, -1.86964065451711\times10^{-8}]$$

Хорошо видно, что оценка ошибки вычисления $x(0.8)$ по Ричардсону совпадает по порядку величины с фактической ошибкой:

```
0.211081834707056-x_exact.subs(t=0.8)
```

$$-2.22778795411216\times10^{-8}$$

В качестве второй таблицы возьмем таблицу 8-го порядка, приведенную у Стоуна за no. 8b-1.

```
P=erk(pr2, N=10, tableau=B)
Q=erk(pr2, N=20, tableau=B)
richardson(P,Q,x,0.8)
```

$$\left[0.211081856984935, 2.39459868056406\times10^{-17}\right]$$

Оценка ошибки по Ричардсону несколько занижена по сравнению с фактической ошибкой:

```
0.211081856984935-x_exact.subs(t=0.8)
```

$$-5.27355936696949\times10^{-16}$$

Диаграмма Ричардсона (рис. 3) вполне проясняет это затруднение: мы практически сразу выходим на ошибку округления, которая имеет порядок 10^{-16}.

Наша реализация метода Рунге–Кутты не зависит от поля $\mathbb{K}$, это позволяет без труда менять число бит, отведенных на одно число и сдвинуть ошибку округления. Разумеется, и элементы таблицы Бутчера, и начальные условия переводятся в это поле. Таким образом, ошибки округления согласованы друг с другом.

Пример 5. Если в методе Рунге–Кутты из примера 4 отвести 300 бит на число, то получится диаграмма Ричардсона (рис. 4), имеющая правильный наклон 8. Для ее построения в функции `erk` была добавлена опция `field=RealField(300)`.

Отмеченный выше вопрос о том, как распорядиться оставшимся произволом в выборе элементов таблицы Бутчера, остается до сих пор нерешенным [19]. В нашем пакете можно попробовать экзотические варианты, напр., взять комплексные значения элементов таблицы, добавив в verb|erk| опцию `field=CC`.

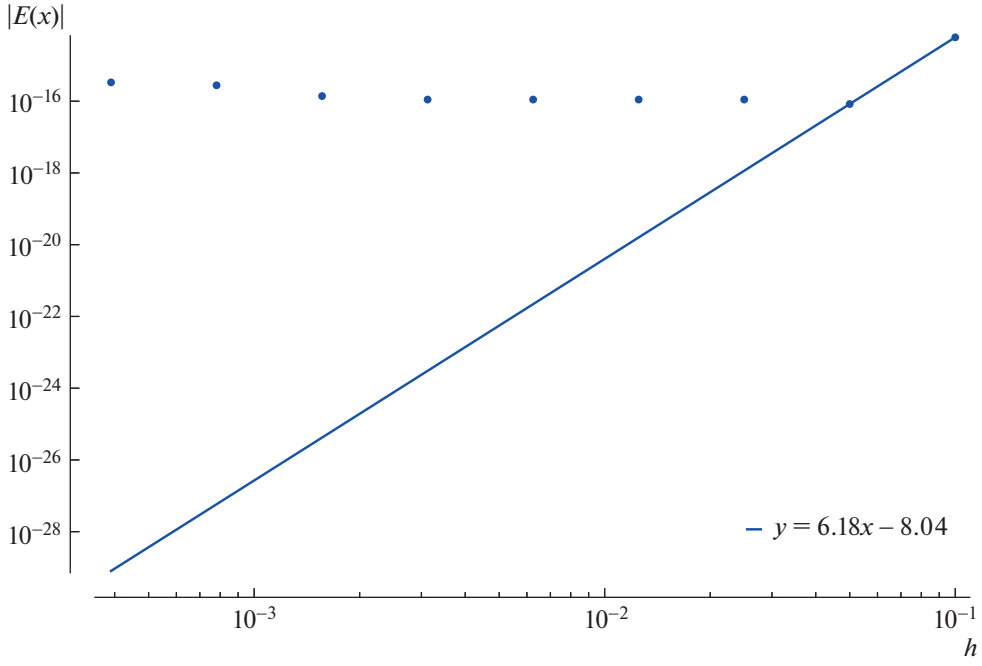

Рис. 3. Диаграмма Ричардсона для значения x_1 при $t = 0.8$ для примера 4.

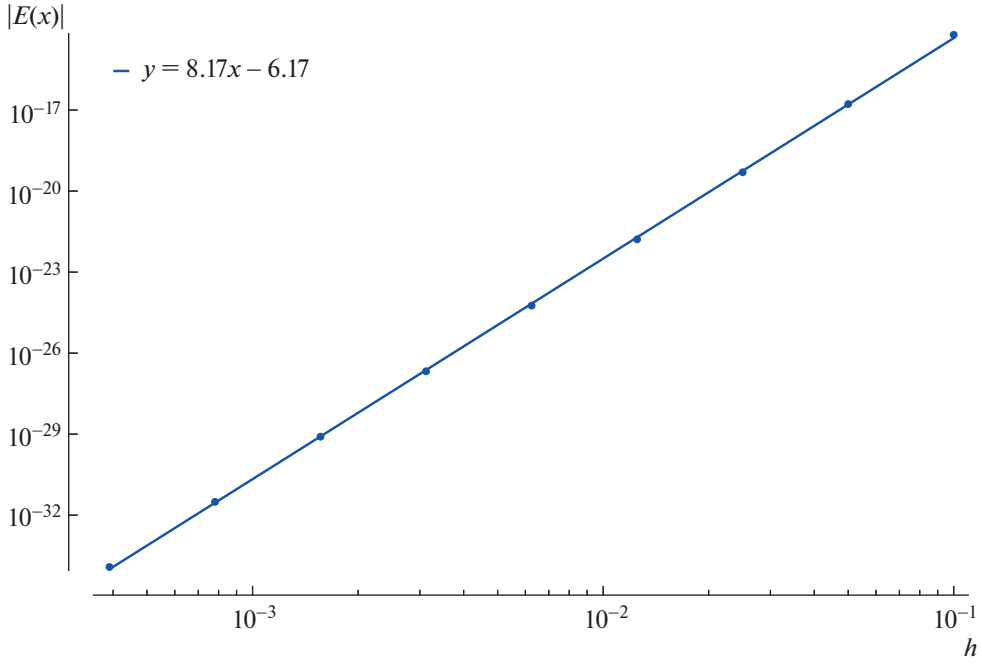

Рис. 4. Диаграмма Ричардсона для значения x_1 при $t = 0.8$ для примера 5.

Пример 6. Таблица

$$\begin{array}{c|c} i & i \\ \hline \frac{1}{2}i+1 & -\frac{1}{2}i \end{array}$$

удовлетворяет системе (5.1) и поэтому задает разностную схему 2-го порядка и в этом смысле ничем не лучше классической схемы средней точки с таблицей

$$\begin{array}{c|c} 1 & 1 \\ \hline \frac{1}{2} & \frac{1}{2} \end{array}$$

Для линейных задач результаты расчетов с вещественной и мнимой таблицами совпадают. Разли-

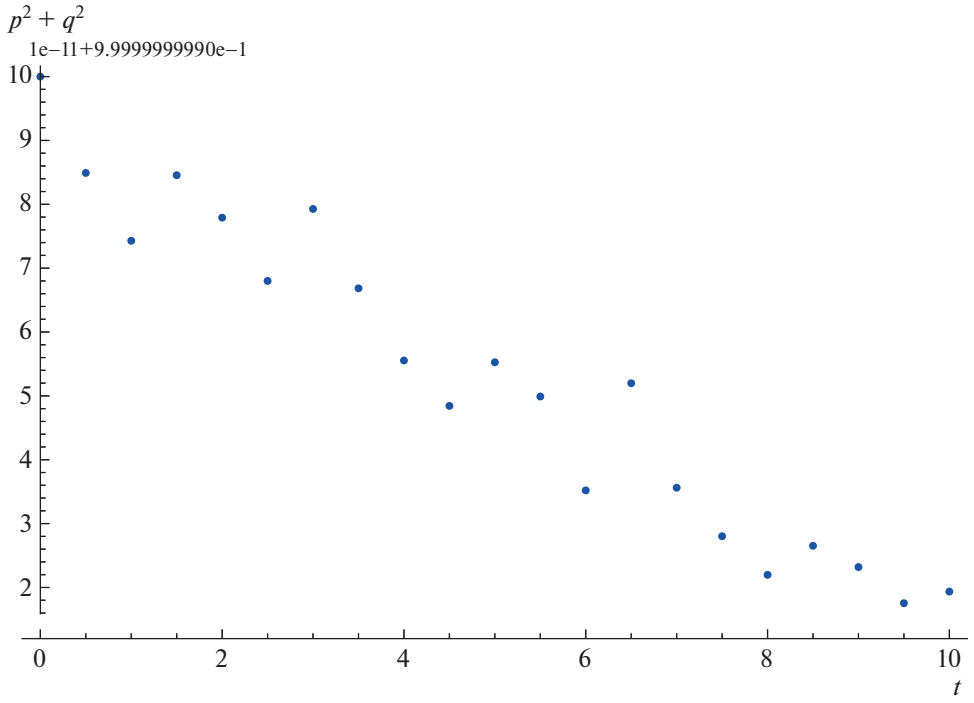

Рис. 5. График зависимости интеграла движения $p^2 + q^2$ осциллятора Якоби от t, вычисленный по схеме средней точки, $N = 20$.

чие проявляется только в нелинейных задачах. Возьмем для примера осциллятор Якоби [27] на отрезке $0 < t < 10$ и сравним значение p при $t = 10$:

```
var("p,q,r,t")
k=1/2
pr3=Initial_problem([p,q,r], [q*r, -p*r, -k^2*p*q], [0,1,1], 10)
  erk(pr3, N = 200, tableau=B, field=CC) .list()[-1] [1]
```

$$0.110646443410733$$

```
erk(pr3, N = 200, tableau=Bi, field=CC) .list()[-1] [1]
```

$$0.111457845209401 - 0.000811190390574542 \cdot i$$

5.3. Неявный метод Рунге–Кутты

Функция `irk` реализует неявный метод Рунге–Кутты, синтаксис тот же, что и у `erk`. Здесь на каждом шаге решается система нелинейных уравнений методом простых итераций. Итерации происходят до тех пор, пока норма разности двух последовательных итераций не оказывается меньше числа $\epsilon > 0$. По умолчанию используется таблица Бутчера

$$\begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array}$$

задающая схему средней точки, и $\epsilon = 10^{-10}$.

Пример 7. Рассмотрим вновь осциллятор Якоби (задача `pr3` из примера 6). Вычислим приближенное решение и построим график зависимости интеграла движения $p^2 + q^2$ от t:

```
P=irk(pr3, N = 20)
P.plot(t,p^2+q^2)
```

В результате получится график, приведенный на рис. 5. Хорошо видно, что рассматриваемый квадратичный интеграл сохраняется с точностью до 10^{-10}. В силу теоремы Купера [7] этот интеграл сохраняется на решениях, полученных по симплектическим схемам Рунге–Кутты, точно. Ошибка округления и параметр ϵ, входящий в критерий остановки итерационного процесса, делают изменение интеграла движения малым, но все же отличным от нуля.

Опция `field`, как и раньше, позволяет менять число бит, отводимых на десятичную дробь, однако теперь ее нужно согласовывать со значением ϵ, которое определяется опцией `eps`, и максимальным числом итераций, которое определяется опцией M.

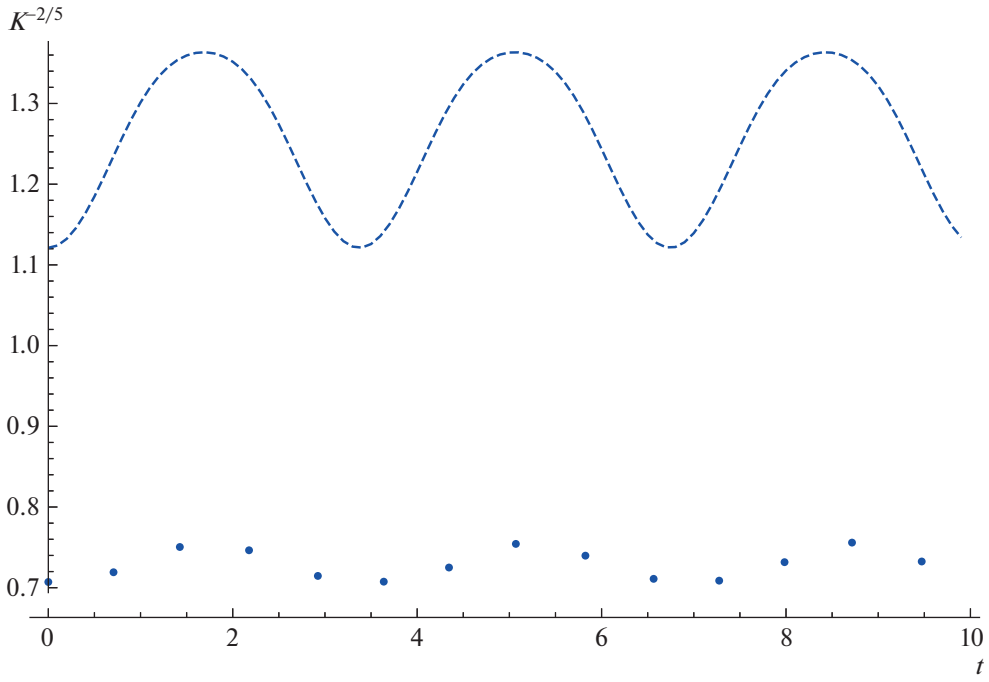

Рис. 6. График изменения шага, использованного для колебаний осциллятора Якоби от t, вычисленный по схеме средней точки, $h = 1$, (сплошная), и кривизна K интегральной кривой (пунктир).

Сходимость метода последовательных итераций накладывает некоторые условия на величину шага. Поэтому в нашей системе реализована версия неявного метода Рунге–Кутты, которая подстраивает шаг под эти условия. Соответствующая функция названа `irk_adaptive`. Подстройка шага мешает заранее узнать число шагов, потребных для достижения конченого значения $t = T$, поэтому вместо опции N, указывающей число отрезков, на которые делится сегмент $[0,T]$, используется опция h, характеризующая длину переменного шага.

Пример 8. Построим график зависимости $p^2 + q^2$ от t по адаптивной схеме:

```
Q=irk_adaptive(pr3, h=1)
Q.plot(t,p^2+q^2)
```

При этом можно посмотреть и на изменение шага

```
Q.plot_dt()
```

График этой зависимости представлен на рис. 6. В [14] рекомендовалось брать шаг адаптивных методов, исходя из формулы

$$\Delta t = \frac{L}{K^{\frac{2}{5}}},$$

где K – кривизна интегральной кривой. Нетрудно видеть, что в данном случае зависимость шага от t повторяет в существенном ход зависимости $K^{-\frac{2}{5}}$ от t.

6. ЗАКЛЮЧЕНИЕ

В настоящей статье представлен оригинальный пакет `fdm`, встраиваемый в систему компьютерной алгебры Sage, при создании которого мы исходили из следующих общих принципов.

• Реализации методов не зависят от поля $(\mathbb{R},\mathbb{C})$ и тем более от числа бит, отведенных на одно число.

• Действия, которые могут быть выполнены аналитически, выполняются аналитически. Начальные задачи рассматриваются как элементы нового класса, в определении которого предусмотрены инструменты для проведения символьных вычислений.

• Численные решения рассматриваются как элементы нового класса, в определении которого предусмотрены инструменты для интерполяции и визуализации.

В тексте рассмотрены основные инструменты для работы с начальными задачами и численными решениями, которые оторваны от конкретных численных методов, и реализация метода Рунге–Кутты. Помимо рассмотренного выше метода Рунге–Кутты, пакет `fdm` содержит реализацию метода Адамса [28, п. 82], примечательного возможностью проведения части вычислений в символьном виде, и метода, основанного на обратимых схемах [10].

Мы надеемся, что созданный инструмент будет удобен для численно-аналитического иссле-

дования разностных схем и со временем станет глубоко интегрирован с алгебраическим инструментарием Sage.

Благодарности. Работа выполнена при финансовой поддержке РНФ (проект № 20-11-20257).

7. ИСТОЧНИК ФИНАНСИРОВАНИЯ

Работа выполнена при финансовой поддержке РНФ (проект № 20-11-20257).

СПИСОК ЛИТЕРАТУРЫ

1. *Runge C., König H.* Vorlesungen über numerisches Rechnen. Springer-Verlag, 2013.
2. SciPy documentation, 2022. Access mode: https://docs.scipy.org.
3. *Ketcheson D.I., bin Waheed U.* A comparison of high order explicit Runge-Kutta, extrapolation, and deferred correction methods in serial and parallel // CAMCoS. 2014. V. 9. № 2. P. 175–200.
4. *Геворкян М.Н.* Конкретные реализации симплектических численных методов // Вестник РУДН. Серия: Математика. Информатика. Физика. 2013. № 3. P. 77–89.
5. *Castillo J.E., Miranda G.F.* Mimetic discretization methods. Chapman and Hall/CRC, 2013.
6. *Da Veiga L.B., Lipnikov K., Manzini G.* The mimetic finite difference method for elliptic problems. Springer, 2014. V. 11.
7. *Hairer E., Wanner G., Lubich Ch.* Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations. Berlin Heidelberg New York : Springer, 2000.
8. On the Quadratization of the Integrals for the Many-Body Problem / Yu Ying, Ali Baddour, Vladimir P. Gerdt et al. // Mathematics. 2021. V. 9. № 24.
9. *Baddour A., Malykh M.* On Difference Schemes for the Many-Body Problem Preserving All Algebraic Integrals // Phys. Part. Nuclei Lett. 2022. V. 19. P. 77–80.
10. *Baddour A., Malykh M., Sevastianov L.* On Periodic Approximate Solutions of Dynamical Systems with Quadratic Right-Hand Side // J. Math. Sci. 2022. V. 261. P. 698–708.
11. *Stein W.A.* Sage Mathematics Software (Version 6.7). The Sage Development Team, 2015. Access mode: http:// www.sagemath.org.
12. *Малых М.Д., Юй Ин* Методика отыскания алгебраических интегралов дифференциальных уравнений первого порядка // Вестник Российского университета дружбы народов. Серия: Математика, информатика, физика. 2018. Т. 26. № 3. С. 285–291.
13. Вычисления на квазиравномерных сетках / Н.Н. Калиткин, А.Б. Альшин, Е.А. Альшина, Б.В. Рогов. Москва : ФИЗМАТЛИТ, 2005. ISBN: 5-9221-0565-5.
14. *Belov A.A., Kalitkin N.N., Poshivaylo I.P.* Geometrically adaptive grids for stiff Cauchy problems // Doklady Mathematics. 2016. V. 93. № 1. P. 112–116.
15. *Belov A.A., Kalitkin N.N.* Nonlinearity Problem in the Numerical Solution of Superstiff Cauchy Problems // Mathematical Models and Computer Simulations. 2016. V. 8. № 6. P. 638–650.
16. Explicit methods for integrating stiff Cauchy problems / A.A. Belov, N.N. Kalitkin, P.E. Bulatov, E.K. Zholkovskii // Doklady Mathematics. 2019. V. 99. № 2. P. 230–234.
17. *Баддур Али, Малых М.Д.* Richardson–Kalitkin method in abstract description // Discrete and Continuous Models and Applied Computational Science. 2021. V. 29. № 3. P. 271–284.
18. Numerical determination of the singularity orderof a system of differential equations / Али Баддур, М.Д. Малых, А.А. Панин, Л.А. Севастьянов // Discrete and Continuous Models and Applied Computational Science. 2020. V. 28. № 1. P. 17–34.
19. *Hairer E., Wanner G., Norsett S.P.* Solving Ordinary Differential Equations I. 3 edition. Springer, 2008.
20. *Yu Ying.* The symbolic problems associated with Runge-Kutta methods and their solving in Sage // Discrete and Continuous Models and Applied Computational Science. 2019. V. 27. № 1. P. 33–41.
21. *Хашин С.И.* Численное решение уравнений Бутчера // Вестник ИвГУ. 2000. № 3. P. 155–164.
22. *Хаммуд Г.М., Хашин С.И.* Шестимерное семейство 6-шаговых методов Рунге–Кутта порядка 5 // Науч. тр. ИвГУ. Математика. 2001. № 4. P. 114–122.
23. *Хашин С.И.* Альтернативная форма уравнений Бутчера // Вестник ИвГУ. 2007. № 3. P. 94–103.
24. *Хашин С.И.* A Symbolic-Numeric Approach to the Solution of the Butcher Equations // Canadian Applied Mathematics Quarterly. 2009. V. 17. № 3. P. 555–569.
25. *Хашин С.И.* Три упрощающих предположения для методов Рунге–Кутта // Вестник ИвГУ. 2012. № 2. С. 142–150.
26. *Stone P.* Maple worksheets on the derivation of Runge-Kutta schemes, 2021. Access mode: http://www.peterstone.name/Maplepgs/ RKcoeff.html.
27. *Сикорский Ю.С.* Элементы теории эллиптических функций с приложениями к механике. М.-Л. : ОНТИ, 1936.
28. *Скарборо Дж.* Численные методы математического анализа. М.-Л.: ГТТИ, 1934.

On Implementation of Numerical Methods for Solving Ordinary Differential Equations in Computer Algebra Systems

© 2023 г. A. Baddour[a], M. M. Gambaryan[a], L. Gonzalez[a], and M. D. Malykh[a,b,#]

[a]*Peoples' Friendship University of Russia, ul. Miklukho-Maklaya 6, Moscow, Russia*

[b]*Joint Institute for Nuclear Research, Dubna, Moscow oblast, 141980 Russia*

[#]*e-mail: malykh_md@pfur.ru*

This paper presents an original package for investigating numerical solutions of ordinary differential equations, which is built in the Sage computer algebra system. This project is focused on a closer integration of numerical and symbolic methods while primarily aiming to create a convenient tool for working with numerical solutions in Sage. The package defines two new classes: initial problems and approximate solutions. The first class defines tools for symbolic computations related to initial problems, while the second class defines tools for interpolating values of symbolic expressions on an approximate solution and estimating the error with the use of the Richardson method. An implementation of the Runge–Kutta method is briefly described, with its main feature being the possibility of working with arbitrary Butcher tableaux and arbitrary numeric fields.

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 59–69

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 512.547.2:530.145.81

ДОПОЛНИТЕЛЬНОСТЬ В КОНЕЧНОЙ КВАНТОВОЙ МЕХАНИКЕ И КОМПЬЮТЕРНЫЕ ВЫЧИСЛЕНИЯ КОМПЛЕМЕНТАРНЫХ НАБЛЮДАЕМЫХ

© 2023 г. В. В. Корняк[a,*]

[a]*Объединенный институт ядерных исследований, 141980 Дубна, Московская область, Россия*

******E-mail: vkornyak@gmail.com*

Поступила в редакцию 17.07.2022 г.
После доработки 17.08.2022 г.
Принята к публикации 30.10.2022 г.

Математическая формулировка принципа дополнительности Бора приводит к понятиям взаимно несмещенных базисов в гильбертовых пространствах и комплементарных квантовых наблюдаемых. Мы рассматриваем связанные с этими понятиями алгебраические структуры и их приложения к конструктивной квантовой механике. Кратко обсуждаются компьютерно-алгебраические подходы к рассматриваемым задачам и приводится алгоритм для решения одной из них.

DOI: 10.31857/S0132347423020115, **EDN:** MFYKPU

1. ВВЕДЕНИЕ

Согласно *принципу дополнительности* (*комплементарности*) Бора для полного описания квантовомеханических явлений неизбежно использование взаимоисключающих, т.е., не допускающих одновременного измерения, наборов данных. Этому принципу, сформулированному Бором на гносеологическом уровне[1], можно придать более конкретный математический смысл с помощью введенного Швингером [2] понятия, для которого впоследствии был предложен ставший общепринятым термин [3] “взаимно несмещенные базисы”.

Два ортонормальных базиса

$$\mathcal{A} = \{|a_0\rangle, |a_1\rangle, \ldots, |a_{N-1}\rangle\}$$

и

$$\mathcal{B} = \{|b_0\rangle, |b_1\rangle, \ldots, |b_{N-1}\rangle\}$$

в N-мерном гильбертовом пространстве называются *взаимно несмещенными*, если вероятности переходов между состояниями, измеренными в разных базисах, одинаковы для любых таких пар состояний, т.е.,

$$|\langle a_k|b_\ell\rangle|^2 = \frac{1}{N} \quad k, \ell = 0, 1, \ldots, N-1.$$

Классическим примером взаимно несмещенных базисов являются базисы собственных состояний операторов координаты $\hat{x}$ и импульса $\hat{p}$ для одномерного движения квантовой частицы. Очевидно, что этот континуальный пример, соответствующий размерности $N = \infty$, выходит за рамки конструктивных теорий. Минимальный, но очень важный базовый пример дают базисы спиновых состояний частицы спина $\frac{1}{2}$ для двух перпендикулярных направлений. В этом случае размерность $N = 2$.

Мы будем называть квантовые наблюдаемые A и B *канонически сопряженными* (или *комплементарными*), если множества их нормализованных собственных векторов образуют взаимно несмещенную пару.

В многочисленных исследованиях понятия квантовой дополнительности, начиная с работ Вейля и Швингера, было показано, что в N-мерном гильбертовом пространстве не может быть (с точностью до унитарной эквивалентности) больше, чем $N + 1$ взаимно сопряженных наблюдаемых. При этом все они конструируются как алгебраические следствия определенной *базовой пары* канонически сопряженных наблюдаемых, одна из которых (мы будем обозначать ее символом X, поскольку при $N = 2$ она аналогична спиновой матрице Паули σ_x) порождается циклической перестановкой N элементов (эти элементы можно отождествить с базисными векторами рассматриваемого N-мерного гильбертова простран-

[1] Предполагается, что вариации принципа дополнительности можно применять в различных областях, где ради полноты описания используются различные, одновременно несовместимые, средства. Например, в статье представлен обзор принципа дополнительности в биологии, психологии и социальных науках.

ства), а другая наблюдаемая Z (аналог матрицы Паули σ_z) получается приведением матрицы X к диагональному виду с помощью преобразования Фурье. Несмотря на эту эквивалентность между наблюдаемыми X и Z (по сути это одна и та же матрица, записанная в двух разных базисах), из формализма квантовой механики следует, что для воспроизведения квантовых интерференций необходимо рассматривать эти наблюдаемые как различные элементы[2].

Тот факт, что канонически сопряженные наблюдаемые по существу происходят из единственной циклической перестановки[3] хорошо согласуется с подходом к квантовой механике, основанном на унитарных представлениях групп перестановок [4–6]. Г. 't Хоофт также предполагает, что на самом глубинном уровне в основе квантовой механики лежат перестановки [7].

Алгебраические соотношения для канонически сопряженных наблюдаемых можно получить конструируя проективные представления или, эквивалентно, центральные расширения абелевых групп. В результате возникают алгебраические структуры, называемые обобщенными алгебрами Клиффорда.

Канонически сопряженные наблюдаемые можно использовать в задачах "квантовой мереологии", т.е. в исследовании конструктивных методов разложения квантовых систем на подсистемы и связанных с такими разложениями задач: вычисление квантовых корреляций и других количественных характеристик взаимодействий между подсистемами, изучение временно́й эволюции этих характеристик и т.д.

Компьютерные вычисления являются эффективным методом исследования рассматриваемых в данной статье проблем. Для построения и исследования систем канонически сопряженных наблюдаемых и связанных с ними задач разрабатываются и реализуются алгоритмы, основанные, главным образом, на методах компьютерной алгебры.

2. КАНОНИЧЕСКИЕ КОММУТАЦИОННЫЕ СООТНОШЕНИЯ

2.1. Каноническое коммутационное соотношение Гейзенберга

В основе квантовой механики лежит каноническое коммутационное соотношение

$$[a,b] = \mathbf{i}\hbar \tag{2.1}$$

между канонически сопряженными наблюдаемыми a и b, представляющими собой эрмитовы операторы. Типичный пример канонически сопряженных наблюдаемых – операторы координаты $\hat{x} = x$ и импульса $\hat{p} = -\mathbf{i}\hbar\dfrac{\partial}{\partial x}$ квантовой частицы, движущейся в одном измерении.

Без соотношений вида (2.1) невозможно описание квантового поведения, в частности, квантовых интерференций. Теорема Стоуна–фон Неймана доказывает единственность соотношений (2.1) для любой квантовой теории в бесконечномерном сепарабельном гильбертовом пространстве.

Канонические коммутационные соотношения (2.1) невозможно реализовать матрицами в пространстве конечной размерности N, поскольку вычисляя, например, след левой и правой частей равенства (2.1) мы получим противоречие:

$$\begin{aligned}&\operatorname{tr}([a,b]) = \operatorname{tr}(ab) - \operatorname{tr}(ba) = 0\\ &\quad\|\\ &\operatorname{tr}(\mathbf{i}\hbar 1_N) = \mathbf{i}\hbar N.\end{aligned}$$

Это означает, что каноническое коммутационное соотношение в форме (2.1) не является фундаментальным, а представляет собой некоторую аппроксимацию.

2.2. Канонически сопряженные наблюдаемые Вейля–Швингера

Герман Вейль [8] построил более фундаментальные коммутационные соотношения, пригодные в N-мерном случае и воспроизводящие соотношение (2.1) в пределе $N \to \infty$.

В квантовой механике наблюдаемые обычно определяются как эрмитовы операторы. Однако Вейль [8], а впоследствии Швингер [2] показали, что для квантовой механики в конечномерных гильбертовых пространствах в качестве наблюдаемых более естественно использовать унитарные операторы вместо эрмитовых. В квантовом формализме главную роль играет разложение гильбертова пространства в прямую сумму ортогональных подпространств, задаваемых собственными векторами оператора, а не конкретные значения собственных чисел, поэтому достаточно, чтобы операторы были *нормальными*[4]. В частности, эрмитовы и унитарные операторы являются нормальными. Внутри множества нормальных операторов они выделяются свойствами спектров: для эрмитовых операторов собственные числа лежат на вещественной оси комплексной плоскости, а для унитарных – на единичной окружности. При необходимости собственные числа операторов, имеющих одинаковые собственные подпространства можно пе-

[2] Необходимость как минимум двух различных базисов (систем координат) для описания физической реальности это своего рода проявление принципа дополнительности.

[3] Заметим, что циклические перестановки это наиболее простые составляющие, на которые разлагается любая перестановка.

[4] Оператор A называется нормальным, если он коммутирует со своим сопряженным: $AA^* = A^*A$.

ресчитать, используя простые функции (например, экспоненты и логарифмы в случае эрмитовых и унитарных операторов).

В книге [8] Вейль показал, что в конечномерном гильбертовом пространстве каноническое коммутационное соотношение имеет вид

$$XZ = \omega ZX \tag{2.2}$$

и доказал его единственность. Здесь $\omega = e^{2\pi i/N}$ – базовый примитивный корень N-й степени из единицы, а унитарные операторы X и Z имеют (с точностью до унитарной эквивалентности) вид

$$X = \begin{pmatrix} 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \\ 1 & 0 & \dots & 0 \end{pmatrix},$$
$$Z = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \omega & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \omega^{N-1} \end{pmatrix}. \tag{2.3}$$

Эти матрицы впервые, в 19-м веке, под названиями “матрица сдвига” и “матрица часов” ввел Дж.Дж. Сильвестр. Очевидно, что

$$X^N = Z^N = 1. \tag{2.4}$$

В минимальной размерности $N = 2$ мы имеем $\omega = -1$, а матрицы X и Z совпадают с матрицами Паули:

$$X = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Поэтому в гильбертовых пространствах произвольной размерности N матрицы X и Z часто называют “обобщенными матрицами Паули”.

Содержательно, матрица X это матрица перестановочного представления генератора циклической перестановки множества из N элементов, а матрица Z это результат приведения матрицы X к диагональному виду унитарным преобразованием:

$$Z = FXF^{-1},$$

где F – матрица преобразования Фурье:

$$F = \frac{1}{\sqrt{N}}(\omega^{-k\ell}), \quad k, \ell = 0,1,\dots,N-1. \tag{2.5}$$

2.3. Группа Вейля–Гейзенберга и унитарный базис Швингера

Пара канонически сопряженных наблюдаемых X и Z является алгебраически полной в том смысле, что из произведений их степеней можно построить полный базис пространства $\mathcal{L}(\mathcal{H})$ линейных операторов в N-мерном гильбертовом пространстве [9].

С помощью соотношений (2.2) и (2.4) произвольное произведение степеней X и Z можно привести к виду

$$Y_{k\ell m} = \omega^k X^\ell Z^m, \quad k, \ell, m = 0,1,\dots,N-1. \tag{2.6}$$

Унитарные операторы (2.6) образуют конечную группу порядка N^3, называемую *группой Вейля–Гейзенберга* или *обобщенной группой Паули*.

Подмножество группы Вейля–Гейзенберга, состоящее из N^2 матриц вида $X^\ell Z^m$, образует базис пространства операторов $\mathcal{L}(\mathcal{H})$, называемый *унитарным базисом Швингера*.

В минимальном случае $N = 2$ группа Вейля–Гейзенберга изоморфна группе кватернионов

$$Q_8 = \{\pm 1, \pm \mathbf{i}, \pm \mathbf{j}, \pm \mathbf{k}\},$$

а унитарный базис Швингера имеет вид,

$$\{1, \sigma_x, \sigma_z, \sigma_x\sigma_z\}$$

или, эквивалентно,

$$\{1, \sigma_x, \sigma_y, \sigma_z\},$$

поскольку матрицу Паули $\sigma_y = \begin{pmatrix} 0 & -\mathbf{i} \\ \mathbf{i} & 0 \end{pmatrix}$ можно представить как $\sigma_y = \mathbf{i}\sigma_x\sigma_z$ – умножение на $\mathbf{i}$ переводит антиэрмитову матрицу $\sigma_x\sigma_z$ в эрмитову и, таким образом, все три спиновые матрицы Паули σ_x, σ_y, σ_z оказываются одновременно и унитарными и эрмитовыми.

3. КАНОНИЧЕСКИ СОПРЯЖЕННЫЕ НАБЛЮДАЕМЫЕ И ВЗАИМНО НЕСМЕЩЕННЫЕ БАЗИСЫ

Рассмотрим два базиса в N-мерном гильбертовом пространстве

1. $\mathcal{B}_o = \{|0\rangle, |1\rangle, \dots, |N-1\rangle\}$ – базис, в котором оператор X действует как циклическая перестановка в соответствии с (2.3) (исходный “онтический базис”).

2. $\mathcal{B}_\varepsilon = \{|\tilde{0}\rangle, |\tilde{1}\rangle, \dots, |\widetilde{N-1}\rangle\}$ – базис, в котором оператор X имеет диагональный вид (“энергетический базис”).

Эти два базиса связаны между собой преобразованием Фурье (2.5). Легко показать, что базисы $\mathcal{B}_o$ и $\mathcal{B}_\varepsilon$ являются взаимно несмещенными:

$$\mathcal{B}_\varepsilon = F\mathcal{B}_o$$
$$\Rightarrow |\tilde{\ell}\rangle = \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\omega^{-k\ell}$$
$$\Rightarrow \langle\tilde{\ell}|k\rangle = \frac{1}{\sqrt{N}}\omega^{k\ell}$$
$$\Rightarrow |\langle\tilde{\ell}|k\rangle|^2 = \frac{1}{N} \quad \square$$

Возникает естественный вопрос: какие еще, помимо X и Z, сопряженные наблюдаемые и ассоциированные с ними взаимно несмещенные базисы возможны в гильбертовом пространстве данной размерности N? Этот вопрос связан непосредственно с проблемой измерений в квантовой механике.

3.1. Квантовая томография

Квантовая томография [9, 10] – это часть квантовой информатики, занимающаяся восстановлением максимума информации о квантовом состоянии с помощью многократных квантовых измерений.

Типичная схема квантовых измерений предполагает разбиение гильбертова пространства на одномерные ортогональные подпространства, соответствующие элементам некоторого ортонормального базиса. Единичное квантовое событие фиксируется датчиком как попадание квантовой системы в одно из подпространств. Проведя достаточно большое количество измерений одного и того же (т.е., “идентично приготовленного”) состояния в данном базисе мы получим набор вероятностей обнаружения системы в каждом из подпространств. Поскольку в силу полноты рассматриваемой схемы измерений сумма всех вероятностей равна единице, в N-мерном пространстве мы получим $N - 1$ независимых параметров, описывающих статистику измерений данного квантового состояния в данном базисе.

Квантовое состояние общего вида описывается матрицей плотности, т.е., эрмитовой матрицей с единичным следом. Легко подсчитать, что для полного описания такой матрицы необходимо $N^2 - 1$ вещественных параметров. Таким образом, для полного восстановления квантового состояния нам необходимо выполнить серию измерений в

$$(N^2 - 1)/(N - 1) = N + 1$$

различных базисах. Наиболее оптимальным для квантовой томографии было бы иметь $N + 1$ взаимно несмещенных базисов, поскольку измерения в таких базисах максимально независимы.

3.2. Максимальные множества взаимно несмещенных базисов

Более детальный анализ множества матриц плотности показывает, что по чисто геометрическим причинам [11] число взаимно несмещенных базисов не может превышать $N + 1$.

Пусть M_N – максимальное число взаимно несмещенных базисов в N-мерном гильбертовом пространстве. Если размерность гильбертова пространства равна степени простого числа, т.е., $N = p^m$, то непосредственным построением было показано, что M_N равно максимально возможному значению $N + 1$. Например, если $N = p$ – простое число, то несложно проверить [12], что собственные базисы унитарных наблюдаемых X, Z, XZ, XZ^2, ..., XZ^{p-1} образуют множество взаимно несмещенных базисов с максимально возможным числом элементов $N + 1$. Аналогичный набор сопряженных унитарных наблюдаемых можно построить и в более общем случае $N = p^m$ с помощью конструкций, использующих вычисления в полях Галуа GF(p^m) [9, 13].

В общем случае размерность можно разложить в произведение элементарных множителей – степеней простых чисел

$$N = p_1^{m_1}\cdots p_k^{m_k}\cdots p_K^{m_K}.$$

В соответствии с этим разложением полное гильбертово пространство можно представить в виде тензорного произведения

$$\mathcal{H}_N = \bigotimes_{k=1}^{K}\mathcal{H}_{N_k},$$

где $N_k = p_k^{m_k}$ – k-й элементарный множитель.

Поскольку у нас имеются в явном виде максимальные множества сопряженных наблюдаемых (назовем эти наблюдаемые локальными) для каждого тензорного множителя $\mathcal{H}_{N_k}$, можно сконструировать взаимно несмещенные базисы пространства $\mathcal{H}_N$ в виде тензорных произведений базисов локальных наблюдаемых. Легко проверить, что если два (глобальных) базиса в пространстве $\mathcal{H}_N$ являются тензорными произведениями взаимно несмещенных локальных базисов, то эти глобальные базисы – взаимно несмещенные. Отсюда следует, что максимальное число взаимно несмещенных глобальных базисов должно удовлетворять условию

$$M_{N=p_1^{m_1}\cdots p_K^{m_K}} \geq \min\{p_1^{m_1},\ldots,p_K^{m_K}\} + 1. \qquad (3.1)$$

Однако задача построения максимального множества взаимно несмещенных базисов в произвольной размерности очень трудна [9] и не решена даже в простейшем случае $N = 2 \times 3$.

4. ПРОЕКТИВНЫЕ ПРЕДСТАВЛЕНИЯ АБЕЛЕВЫХ ГРУПП И ОБОБЩЕННЫЕ АЛГЕБРЫ КЛИФФОРДА

Проективные представления групп являются одним из центральных элементов квантового формализма. В частности, коммутационное соотношение (2.2) Вейль получил с помощью проективного представления абелевой группы, являющейся прямым произведением двух циклических групп [8]. Проективные представления абелевых групп можно описать в рамках общей конструкции, называемой *обобщенными алгебрами Клиффорда* [14].

4.1. Проективные представления

Правило умножения для операторов проективного (лучевого) представления R группы G в линейном пространстве над полем $\mathcal{F}$ имеет вид

$$R(g_1)R(g_2) = c(g_1, g_2)R(g_1g_2), \tag{4.1}$$

где $g_1, g_2 \in G$, а функция от двух аргументов

$$c : G \times G \to \mathcal{F}^* \equiv \mathcal{F}\backslash\{0\} \tag{4.2}$$

должна удовлетворять тождеству

$$c(g_1, g_2)c(g_1g_2, g_3) = c(g_1, g_2g_3)c(g_2g_3),$$

которое можно получить, если привести произведение $R(g_1)R(g_2)R(g_3)$ к $R(g_1g_2g_3)$ двумя разными путями используя (4.1).

Матрицы представления R определены с точностью до умножений на произвольные ненулевые элементы поля, т.е. $R(g)$ – представитель класса эквивалентности вида

$$R(g) \sim b(g)R(g), \tag{4.3}$$

где $b : G \to \mathcal{F}^*$ – произвольная функция на группе. Изменение выбора представителей приводит к замене $c(g_1, g_2) \to c_b(g_1, g_2)c(g_1, g_2)$ в формуле (4.1), где

$$c_b(g_1, g_2) = \frac{b(g_1)b(g_2)}{b(g_1g_2)}. \tag{4.4}$$

Функции (4.2) образуют коммутативную группу $Z^2(G, \mathcal{F}^*)$, называемую группой двумерных коциклов (2-коциклов). Функции вида (4.4), называемые *когранцами*, образуют подгруппу $B^2(G, \mathcal{F}^*)$ группы $Z^2(G, \mathcal{F}^*)$. Фактор-группа

$$H^2(G, \mathcal{F}^*) = Z^2(G, \mathcal{F}^*)/B^2(G, \mathcal{F}^*)$$

называется двумерной группой *когомологий* группы G с коэффициентами в $\mathcal{F}^*$.

Проективное представление R однозначно определяет некоторый класс $[c_R] \in H^2(G, \mathcal{F}^*)$. Любой коцикл $c(g_1, g_2) \in [c_R]$ называется *мультипликатором* [15] проективного представления R. Мультипликаторы, имеющие вид (4.4), принадлежат классу $[1 \in \mathcal{F}^*]$ и называются тривиальными. Проективное представление с тривиальным мультипликатором эквивалентно обычному линейному представлению.

Не всякая группа имеет нетривиальные проективные представления. Например, циклическая группа порядка n порождается одним элементом и имеет вид $G = \{1, g, g^2, ..., g^{n-1}\}$. Для проективного представления такой группы должно выполняться равенство $R(g)^n = a1$, где a – некоторый скаляр $a \in \mathcal{F}^*$. Если $a \neq 1$, то, воспользовавшись эквивалентностью (4.3) и сделав замену $R(g) \to \to a^{1/n}R(g)$, мы получим равенство $R(g)^n = 1$, показывающее, что любое проективное представление циклической группы эквивалентно обычному линейному представлению.

Наименьшей группой, имеющей нетривиальные проективные представления, является *четверная группа* Клейна $K_4 \simeq \mathbb{Z}_2 \times \mathbb{Z}_2$. Проективные представления даже в этом минимальном случае порождают богатый набор существенных для физики структур: описание частиц спина $\frac{1}{2}$; кватернионы и, в частности, описание трехмерных вращений; матрицы Паули и состоящие из них матрицы Дирака, лежащие в основе уравнения Дирака и т.д.

4.2. Обобщенные алгебры Клиффорда

Обобщенной алгеброй Клиффорда называется алгебра, порождаемая элементами $e_1, e_2, ..., e_n$, удовлетворяющими соотношениям

$$e_i^{N_i} = 1, \tag{4.5}$$

$$e_ie_j = \omega_{ij}e_je_i, \tag{4.6}$$

$$\omega_{ij}^{N_i} = \omega_{ij}^{N_j} = 1, \tag{4.7}$$

$$\omega_{ij}e_k = e_k\omega_{ij}, \quad \omega_{ij}\omega_{k\ell} = \omega_{k\ell}\omega_{ij}, \tag{4.8}$$

где $i, j, k, \ell = 1, 2, ..., n$, N_i – положительные целые числа.

Для любых матричных представлений, имеющих смысл в физических приложениях, достаточно выбрать элементы ω_{ij} в виде корней из единицы

$$\omega_{ij} = \omega_{ji}^{-1} = e^{2\pi i s_{ij}/N_{ij}}, \tag{4.9}$$

где s_{ij} – целые числа, $N_{ij} = \gcd(N_i, N_j)$. При таком выборе необходимость в соотношениях (4.8) отпадает[5].

[5] Обычной алгебре Клиффорда, которая определяется с помощью разложения заданной квадратичной формы в n-мерном пространстве в произведение линейных множителей, соответствует случай корней из единицы 2-й степени, т.е. $\omega_{ij} = -1$ и соотношения (4.6) принимают вид $e_ie_j = -e_je_i$.

Введя общий период для всех корней из единицы $N = \mathrm{lcm}(\{N_{ij}\})$ можно несколько унифицировать (4.9):

$$\omega_{ij} = \mathrm{e}^{2\pi \mathbf{i} t_{ij}/N}, \quad t_{ij} = -t_{ji}.$$

Таким образом, обобщенная алгебра Клиффорда задается множеством периодов

$$N_1, N_2, \ldots, N_n$$

для порождающих элементов и кососимметрической целочисленной матрицей

$$T = \begin{pmatrix} 0 & t_{12} & \ldots & t_{1n} \\ -t_{12} & 0 & \ldots & t_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -t_{1n} & -t_{2n} & \ldots & 0 \end{pmatrix}$$

4.3. Проективные представления абелевых групп

Произвольную (конечную) абелеву группу можно представить в виде прямого произведения циклических групп

$$G = \mathbb{Z}_{N_1} \times \mathbb{Z}_{N_2} \times \ldots \times \mathbb{Z}_{N_n}. \tag{4.10}$$

Замечание. Циклическая группа разлагается в прямое произведение подгрупп тогда и только тогда, когда порядки сомножителей взаимно просты, например,

$$\mathbb{Z}_{k\ell} \simeq \mathbb{Z}_k \times \mathbb{Z}_\ell \Leftrightarrow \gcd(k, \ell) = 1.$$

Этот факт позволяет получить классификацию всех абелевых групп с точностью до изоморфизма:

$$G_{P,M} = \mathbb{Z}_{p_1^{m_1}} \times \mathbb{Z}_{p_2^{m_2}} \times \ldots \times \mathbb{Z}_{p_n^{m_n}},$$

где $P = \{p_1, \ldots, p_n\}$ – множество простых чисел (необязательно различных!), $M = \{m_1, \ldots, m_n\}$ – множество положительных целых чисел. Мы однако не будем предполагать, что в разложении (4.10) числа $N_1, \ldots, N_n$ – степени простых чисел.

Абелеву группу (4.10) можно задать с помощью соотношений

$$c_i^{N_i} = 1, \quad c_i c_j = c_j c_i, \quad i, j = 1, 2, \ldots, n, \tag{4.11}$$

где c_i обозначает образующую множителя $\mathbb{Z}_{N_i}$ в разложении (4.10). Подробное вычисление мультипликаторов проективного представления для G исходя из соотношений (4.11) приведено в [14]. Результатом вычислений являются соотношения для проективных образов $e_i = R(c_i)$, полностью совпадающие с соотношениями (4.5)–(4.8) для обобщенных алгебр Клиффорда.

5. КАНОНИЧЕСКИ СОПРЯЖЕННЫЕ НАБЛЮДАЕМЫЕ В КОНЕЧНОЙ КВАНТОВОЙ МЕХАНИКЕ

Канонически сопряженные наблюдаемые Вейля–Швингера возникают как необходимое следствие возможности описывать квантовые интерференции в конечномерном гильбертовом пространстве.

Более того, в их основе лежит циклическая перестановка элементов конечного множества. Фактически из требования конечномерности пространства следует конечность множества состояний, что согласуется с идеологией версии квантовой механики, основанной на перестановках конечных множеств.

Соответствующая модификация квантового формализма [4–6], которую мы для краткости называем конечной квантовой механикой, обсуждается в [16].

Г. 't Хоофт также предполагает, что перестановки лежат в основе квантовой эволюции ([7], стр. 66):

> *We postulate the existence of an ontological basis. It is an orthonormal basis of Hilbert space that is truly superior to the basis choices that we are familiar with. In terms of an ontological basis, the evolution operator for a sufficiently fine mesh of time variables, does nothing more than permute the states.*

5.1. Эволюция замкнутой квантовой системы

Один из главных постулатов квантовой механики утверждает, что эволюция замкнутой квантовой системы описывается унитарным преобразованием.

В стандартной (континуальной) квантовой механике эволюция описывается однопараметрической (поэтому циклической) группой унитарных преобразований $U^t = \mathrm{e}^{-\frac{\mathbf{i}}{\hbar}Ht}$. Инфинитезимальным порождающим элементом этой группы является гамильтониан H, а параметром, нумерующим элементы группы – непрерывное время $t \in$. Эволюция из заданного состояния $|\Psi_0\rangle$ в состояние $|\Psi_t\rangle$ описывается уравнением

$$|\Psi_t\rangle = U^t |\Psi_0\rangle,$$

инфинитезимальной версией которого, получаемой дифференцированием по времени, является уравнение Шредингера

$$\mathbf{i}\hbar \frac{\partial |\Psi_t\rangle}{\partial t} = H|\Psi_t\rangle.$$

В конечной квантовой механике мы предполагаем, что на самом глубоком уровне описания су-

ществует конечное множество первичных объектов

$$\Omega \simeq \{0,1,\ldots,\mathcal{N}-1\}, \tag{5.1}$$

которые мы, следуя терминологии 'т Хоофта, будем называть *онтическими*[6]. Любое обратимое преобразование этого множества является перестановкой из группы $\mathrm{Sym}(\Omega)$. Мы можем отождествить множество Ω с ортонормальным базисом некоторого $\mathcal{N}$-мерного гильбертова пространства $\mathcal{H}_{\mathcal{N}}$. Эти пространство и базис также будем называть *онтическими*.

Любую перестановку $g \in \mathrm{Sym}(\Omega)$ можно представить в пространстве $\mathcal{H}_{\mathcal{N}}$ в виде унитарного оператора

$$U = \mathcal{M}(g), \tag{5.2}$$

компоненты которого имеют вид

$$\mathcal{M}(g)_{k,\ell} = \delta_{kg,\ell}, \quad k,\ell \in \Omega.$$

Эволюция замкнутой квантовой системы в конечной квантовой механике описывается циклической группой U^t, где $t \in$ – дискретное время, а порождающим элементом является унитарный оператор (5.2). Любую перестановку $g \in \mathrm{Sym}(\Omega)$ можно разложить в произведения непересекающихся циклов

$$\begin{gathered} g = (c_1)\ldots(c_k)\ldots(c_K), \\ c_k^{L_k} = 1, \quad \sum_{k=1}^{K} L_k = \mathcal{N}, \end{gathered} \tag{5.3}$$

где L_k – длина цикла c_k, а символ $\mathbf{1}$ обозначает тождественную перестановку соответствующей длины.

В соответствии с разложением (5.3) оператор (5.2), порождающий эволюцию, приобретает вид

$$U = \begin{pmatrix} \mathcal{M}(c_1) & 0 & \ldots & 0 \\ 0 & \mathcal{M}(c_2) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \mathcal{M}(c_K) \end{pmatrix}.$$

Это означает, что онтическое множество разбивается в объединение непересекающихся подмножеств

$$\Omega = \Omega_1 \sqcup \Omega_2 \sqcup \ldots \sqcup \Omega_K, \tag{5.4}$$

в каждом из которых действует циклическая перестановка, а онтическое гильбертово пространство разбивается в прямую сумму инвариантных подпространств

$$\mathcal{H}_{\mathcal{N}} = \mathcal{H}_{L_1} \oplus \mathcal{H}_{L_2} \oplus \ldots \oplus \mathcal{H}_{L_K},$$

унитарная эволюция в каждом из которых задается оператором вида (2.3). Поэтому мы можем рассматривать эволюцию в каждом из подмножеств в разложении (5.4) независимо от остальных подмножеств, причем эта эволюция будет простой циклической перестановкой.

Чтобы избежать громоздких обозначений для выделения одного из подмножеств в разбиении (5.4), мы будем рассматривать циклическую эволюцию длины $\mathcal{N}$ всего онтического множества.

5.2. Канонически сопряженные наблюдаемые и взаимно несмещенные базисы

Онтический базис, в соответствии с (5.1), имеет вид

$$\mathcal{B}_o = \{|0\rangle, |1\rangle, \ldots, |\mathcal{N}-1\rangle\}. \tag{5.5}$$

Циклическая эволюция порождается онтической канонической наблюдаемой $\mathcal{X}$, действие которой на элементы онтического базиса (5.5) имеет вид

$$\mathcal{X}|k\rangle = |k-1(\mathrm{mod}\,\mathcal{N})\rangle.$$

Нормированные собственные векторы наблюдаемой $\mathcal{X}$ образуют ортонормальный базис

$$\mathcal{B}_\varepsilon = \{|\tilde{0}\rangle, |\tilde{1}\rangle, \ldots, |\widetilde{\mathcal{N}-1}\rangle\}, \tag{5.6}$$

который мы называем *энергетическим*[7].

В энергетическом базисе, переход к которому от онтического осуществляется преобразованием Фурье, оператор $\mathcal{X}$ имеет вид

$$\mathcal{Z} = \mathrm{diag}(1, \omega, \omega^2, \ldots, \omega^{\mathcal{N}-1}),$$

где $\omega = e^{2\pi i/\mathcal{N}}$ – базовый примитивный корень из единицы степени $\mathcal{N}$.

В разделе 3 показано, что базисы (5.5) и (5.6) являются взаимно несмещенными, а операторы $\mathcal{X}$ и $\mathcal{Z}$ образуют базовую пару канонически сопряженных наблюдаемых.

5.3. Квантовая мереология в конечной квантовой механике

Квантовая мереология изучает взаимосвязи подсистем составной квантовой системы между собой и системой в целом. Заданное множество (локальных) квантовых систем легко объединить согласованным с правилами квантовой механики способом в (глобальную) квантовую систему, для которой локальные системы являются подсистемами. Более важной, интересной и трудной является обратная задача: разложить заданную квантовую систему на подсистемы наиболее разумным (в соответствии с определенными критериями) способом.

[6] 'т Хоофт использует слово *ontic* как сокращение для *ontological*.

[7] Мы пользуемся этим термином поскольку собственные значения оператора $\mathcal{X}$ представляют собой экспоненты частот, которые пропорциональны энергиям согласно формуле Планка $E = h\nu$, считающейся справедливой для периодического процесса любой природы.

Гильбертово пространство глобальной системы является тензорным произведением локальных пространств. Легко показать, что между базисом глобального пространства и набором локальных базисов существует взаимно однозначное соответствие, т.е., не только можно построить глобальный базис из локальных, но можно однозначно восстановить все локальные базисы исходя из глобального [17–19]. Поскольку все базисы гильбертова пространства лежат на одной орбите группы унитарных преобразований, конкретное разложение глобальной квантовой системы на подсистемы можно получить, задав некоторое разложение размерности глобального пространства на множители и зафиксировав некоторый унитарный оператор в глобальном пространстве. После этого остальные характеристики разложения (квантовые корреляции, энергии взаимодействия и т.д.) вычисляются стандартными способами.

Выбрав в качестве глобального пространства онтическое пространство $\mathcal{H}_{\mathcal{N}}$, в качестве разложения размерности – разложение на элементарные множители

$$\mathcal{N} = p_1^{m_1} p_2^{m_2} \ldots p_k^{m_k} \ldots p_K^{m_K}, \tag{5.7}$$

где все p_k – различные простые числа, а в качестве унитарного оператора, фиксирующего глобальный базис – каноническую наблюдаемую $\mathcal{X}$, мы получим следующее разложение для гильбертова пространства

$$\mathcal{H}_{\mathcal{N}} = \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \ldots \otimes \mathcal{H}_K,$$

где $\dim\mathcal{H}_k = p_k^{m_k}$.

Это разложение замечательно тем, что для него глобальные наблюдаемые являются тензорными произведениями локальных наблюдаемых:

$$\begin{aligned} \mathcal{X} &= X_1 \otimes X_2 \otimes \ldots \otimes X_K, \\ \mathcal{Z} &= Z_1 \otimes Z_2 \otimes \ldots \otimes Z_K. \end{aligned} \tag{5.8}$$

Заметим, что эти равенства невозможны при несоблюдении требования различия простых чисел в разложении (5.7).

“Логарифмированием” (5.8) можно получить выражение, связывающее глобальный гамильтониан с локальными

$$\begin{aligned} \mathrm{H} = H_1 \otimes 1_2 \otimes \ldots \otimes 1_K + 1_1 \otimes H_2 \otimes \ldots \otimes 1_K \\ + \ldots + 1_1 \otimes 1_2 \otimes \ldots \otimes H_K, \end{aligned} \tag{5.9}$$

где 1_k – единичные матрицы соответствующих размерностей. Отметим специальный вид формулы (5.9): глобальный гамильтониан равен сумме гамильтонианов подсистем – “энергия системы равна сумме энергий подсистем”. При разложении квантовой системы на подсистемы “наугад” формула аналогичная (5.9) будет, как правило, содержать дополнительно гамильтонианы взаимодействий между подсистемами.

6. ВЫЧИСЛИТЕЛЬНЫЕ ЗАДАЧИ

Для решения многочисленных задач возникающих при исследовании квантовых моделей с использованием обсуждаемых в данной статье методов, мы разрабатываем и реализуем (преимущественно на языке Си) соответствующие компьютерно-алгебраические алгоритмы.

Эти алгоритмы, среди прочего, включают

• средства работы с корнями из единицы. Несмотря на то, что обычно используемое выражение для n-го примитивного корня из единицы, $\omega_n = e^{2\pi i/n}$, содержит два трансцендентных числа, **e** и π, на самом деле ω_n – *целое алгебраическое число*, являющееся корнем n-го циклотомического полинома $\Phi_n(x)$. Поэтому вычисления с корнями из единицы сводятся к несложной арифметике с полиномами от одной переменной.

• вычисления с полями Галуа $\mathrm{GF}(p^m)$. Элементы таких полей используются для нумерации базисных элементов гильбертовых пространств размерности p^m.

• вычисления с матрицами из группы Вейля–Гейзенберга (2.6).

Положительным с вычислительной точки зрения свойством этих матриц является то, что они содержат ровно N (а не N^2) ненулевых элементов в N-мерном случае.

В качестве примера рассмотрим задачу построения максимальных множеств взаимно несмещенных базисов в гильбертовом пространстве заданной размерности N. Для решения этой задачи, помимо теоретических исследований, используется множество вычислительных средств, таких как компьютерная алгебра, численные методы и моделирование Монте-Карло [20–23]. Задача полностью решена для размерностей вида $N = p^m$, т.е. для степеней простого числа. В этом случае с помощью унитарных матриц из базиса Швингера в явном виде построены $N + 1$ – т.е. максимально возможное априори число – взаимно несмещенных базисов. Для составных размерностей $N = 6, 10, 12, \ldots$ задача не решена полностью ни в одном случае. Относительно случая $N = 6$, который, естественно, подвергся наиболее интенсивному изучению (см., например, [21]), имеется твердое, хотя и не доказанное, убеждение, что максимальное число взаимно несмещенных базисов в этой размерности равно трем: $M_{N=6} = 3$.

Обычно алгоритмы поиска взаимно несмещенных базисов основаны на работе с унитарными матрицами из группы Вейля–Гейзенберга (2.6). Мы предлагаем здесь алгоритм, в основе которого лежит “фазовая калибровочная инвариантность” онтического базиса, дающая возможность построить полное множество неэквивалентных комплементарных партнеров онтического базиса.

6.1. Комбинаторный алгоритм поиска взаимно несмещенных базисов

Пусть $\mathcal{B}_o = \{|0\rangle, ..., |k\rangle, ..., |N-1\rangle\}$ – онтический базис. Калибровочное преобразование этого базиса, сводящееся к умножению его элементов на произвольные фазовые множители, приводит к ортонормальному базису

$$\mathcal{B}_{o,\boldsymbol{\lambda}} = \left\{ e^{\frac{2\pi i}{N}\lambda_0}|0\rangle, ..., e^{\frac{2\pi i}{N}\lambda_k}|k\rangle, ..., e^{\frac{2\pi i}{N}\lambda_{N-1}}|N-1\rangle \right\},$$

где мы используем обозначение

$$\boldsymbol{\lambda} = \{\lambda_0, ..., \lambda_k, ..., \lambda_{N-1}\}.$$

Базис $\mathcal{B}_{o,\boldsymbol{\lambda}}$ физически неотличим от исходного базиса $\mathcal{B}_o$, поскольку фигурирующие в квантовых измерениях проекторы в базисные подпространства не меняются:

$$|k\rangle\langle k| \to e^{\frac{2\pi i}{N}\lambda_k}|k\rangle\langle k|e^{-\frac{2\pi i}{N}\lambda_k} = |k\rangle\langle k|.$$

Однако переход к комплементарному базису $\mathcal{B}_{\boldsymbol{\lambda}} = F\mathcal{B}_{o,\boldsymbol{\lambda}}$, ℓ-й элемент которого имеет вид

$$|\boldsymbol{\lambda}, \ell\rangle = \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\omega^{-k\ell}e^{\frac{2\pi i}{N}\lambda_k},$$

приводит к физически наблюдаемым последствиям, поскольку в проекторах базиса $\mathcal{B}_{\lambda}$ появляются физически значимые элементы, зависящие от $\boldsymbol{\lambda}$:

$$|\boldsymbol{\lambda}, \ell\rangle\langle\boldsymbol{\lambda}, \ell| = \frac{1}{N}\sum_{k,k'=0}^{N-1}|k\rangle\langle k'|\omega^{(k'-k)\ell}e^{\boxed{\frac{2\pi i}{N}(\lambda_k-\lambda_{k'})}}.$$

Это один из примеров проявления квантовой дополнительности.

Выражение для $|\boldsymbol{\lambda}, \ell\rangle$ можно переписать в виде

$$|\boldsymbol{\lambda}, \ell\rangle = \frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}|k\rangle\omega^{-k\ell+\lambda_k}. \qquad (6.1)$$

Если размерность N – нечетное число, то все элементы группы Вейля–Гейзенберга (2.6) имеют период N, однако в четной размерности некоторые элементы этой группы имеют период $2N$. Из (6.1) видно, что обеспечить должную периодичность можно, предположив, что λ_k – целые числа в нечетной размерности и полуцелые – в четной. Четная размерность требует дополнительного внимания при программировании алгоритма, но для простоты изложения основных идей алгоритма мы будем далее предполагать, что N – нечетное число.

Каждый базис $\mathcal{B}_{\lambda}$, нумеруемый элементом $\boldsymbol{\lambda}$ множества $\Lambda = \{0, 1, ..., N-1\}^N$, комплементарен онтическому базису $\mathcal{B}_o$, что видно из скалярного произведения базисных элементов $|k\rangle \in \mathcal{B}_o$ и $|\boldsymbol{\lambda}, \ell\rangle \in \in \mathcal{B}_{\boldsymbol{\lambda}}$

$$\langle k \mid \boldsymbol{\lambda}, \ell\rangle = \frac{1}{\sqrt{N}}\omega^{-k\ell+\lambda_k} \Rightarrow |\langle k \mid \boldsymbol{\lambda}, \ell\rangle|^2 = \frac{1}{N}.$$

Пусть $V_\Lambda = \{\mathcal{B}_{\boldsymbol{\lambda}} : \lambda \in \Lambda\}$ – множество всех базисов, комплементарных к онтическому. Эти базисы индексируются элементами множества Λ.

Мы можем интерпретировать множество

$$V = \{\mathcal{B}_o\} \cup V_\Lambda$$

как множество вершин графа, ребрами которого являются комплементарные (т.е., взаимно несмещенные) пары базисов и, таким образом, проблема сводится к классической задаче теории графов – поиску *клик*[8] в графе. Максимальные множества взаимно несмещенных базисов являются кликами максимального размера.

Учитывая фундаментальное значение канонической пары комплементарных наблюдаемых X и Z, а также для уменьшения объема вычислений мы потребуем, чтобы искомая клика обязательно содержала эту пару.

Наблюдаемая X соответствует вершине $\mathcal{B}_o$, которую можно сразу исключить из рассмотрения, поскольку она соединена ребрами со всеми остальными вершинами и подключается автоматически к любой клике, найденной в графе V_Λ.

Наблюдаемой Z соответствует "энергетический базис"

$$\mathcal{B}_\varepsilon \equiv \mathcal{B}_{\mathbf{0}} \in V_\Lambda,$$

где $\mathbf{0} = \{0, 0, ..., 0\} \in \Lambda$.

Первым шагом алгоритма является сканирование множества $\Lambda\backslash\{\mathbf{0}\}$ с выделением множества вершин $V_{\mathbf{0}}$, смежных с вершиной $\mathcal{B}_o$. Ввиду большого размера множества $\Lambda\backslash\{0\}$, это наиболее трудоемкая часть вычислений, вносящая основной вклад в затраты времени. Например, в размерности $N = 11$ мы имеем

$$|\Lambda\backslash\{0\}| = 285311670610 \approx 2.9\times10^{11}.$$

При этом, множество смежных с $\mathcal{B}_o$ вершин, получаемое в результате работы первого шага, имеет небольшой размер, например, $|V_{\mathbf{0}}| = 1210$ для $N = 11$.

Для завершающего шага вычислений – поиску клик в графе $V_{\mathbf{0}}$, мы используем алгоритм Брона–Кербоша [24, 25]. Этот алгоритм является одним из самых эффективных алгоритмов поиска клик[9] и считается стандартным для решения этой задачи.

[8] Кликой называется полный подграф неориентированного графа, не содержащийся в большем полном подграфе.

[9] После работы Брона и Кербоша появился ряд конкурирующих алгоритмов [26–29], но ввиду некритичности этой части наших вычислений, вряд ли целесообразно заниматься сравнительным изучением различных алгоритмов.

Основной процедурой нижнего уровня является определение взаимной несмещенности базисов (смежности вершин графов). Эта процедура сводится к вычислению и анализу выражений $\langle \boldsymbol{\mu}, m|\boldsymbol{\lambda}, \ell\rangle$. Технически удобно отбросить коэффициенты нормировки (степени выражения $\frac{1}{\sqrt{N}}$) и иметь дело с полиномами от корней из единицы. Тогда бóльшая часть вычислений сводится к целочисленной модулярной арифметике и только для окончательного приведения выражений по модулю циклотомического полинома $\Phi_N(\omega)$ применяются методы полиномиальной алгебры.

Проверка несмещенности сводится к следующему. В результате вычисления скалярного произведения $\langle \boldsymbol{\mu}, m|\boldsymbol{\lambda}, \ell\rangle$ возникает полином вида

$$A(\omega) = \sum_{k=0}^{N-1} \omega^{k(m-\ell)+\lambda_k-\mu_k}.$$

Если после редукции по $\Phi_N(\omega)$ квадрата модуля этого полинома выполняется равенство

$$|A(\omega)|^2 = N,$$

то проверка продолжается на другой комбинации базисных элементов, в противном случае проверка прекращается и сообщается, что базисы не являются взаимно несмещенными (отсутствует ребро между парой вершин).

С помощью реализации этого алгоритма на языке Си мы выполнили поиск максимальных множеств взаимно несмещенных базисов для всех размерностей до $N = 11$ включительно.

В нетривиальных размерностях $N = 6$ и $N = 10$, в согласии с результатами работ других авторов, мы получили $M_{N=6} = M_{N=10} = 3$, т.е., минимальные значения, гарантированные неравенством (3.1).

Вычисления проводились на персональном компьютере с процессором 3.3GHz Intel Core i3 2120. Приведем времена вычислений, T_N, в секундах для размерностей $8 \le N \le 11$:

$$T_8 = 4.7, \quad T_9 = 155, \quad T_{10} = 5012,$$
$$T_{11} = 169278.$$

Имеются очевидные возможности существенного улучшения алгоритма и реализации, поэтому есть определенная надежда “добраться” до следующих нетривиальных размерностей $N = 12$, 14, 15.

СПИСОК ЛИТЕРАТУРЫ

1. *Pattee H.H.* The complementarity principle in biological and social structures // Journal of Social and Biological Structures. 1978. V. 1. № 2. P. 191–200. Access mode: https://www.sciencedirect.com/science/article/pii/S0140175078800074.
2. *Schwinger Julian*. Unitary Operator Bases // Proc Natl Acad Sci U S A. 1960. Apr. V. 46. № 4. P. 570–579. Access mode: https://www.pnas.org/doi/abs/10.1073/pnas.46.4.570.
3. *Wootters William K., Fields Brian D.* Optimal state-determination by mutually unbiased measurements // Annals of Physics. 1989. May. V. 191. № 2. P. 363–381.
4. *Kornyak Vladimir V.* Quantum models based on finite groups // Journal of Physics: Conference Series. 2018. Feb. V. 965. P. 012023. Access mode: https://doi.org/10.1088/1742-6596/965/1/012023
5. *Kornyak Vladimir V.* Modeling Quantum Behavior in the Framework of Permutation Groups // EPJ Web of Conferences. 2018. V. 173. P. 01007. Access mode: https://doi.org/10.1051/epjconf/201817301007
6. *Kornyak Vladimir V.* Mathematical Modeling of Finite Quantum Systems // Lect. Notes Comput. Sci. 2012. V. 7125. P. 79–93. 1107.5675.
7. *’t Hooft Gerard*. The Cellular Automaton Interpretation of Quantum Mechanics. Fundamental Theories of Physics. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-82314-0. Access mode: https://doi.org/10.1007/978-3-319-41285-6.
8. *Вейль Герман*. Теория групп и квантовая механика. Москва: “Наука”, 1986. С. 496.
9. On Mutually Unbiased Bases / Durt Thomas, Englert Berthold-Georg, Bengtsson Ingemar, and Z˙ yczkowski Karol // International journal of quantum information. 2010. V. 8. № 4. P. 535–640.
10. *D’Ariano G.* Mauro, Paris Matteo G.A., *Sacchi Massimiliano F.* Quantum tomography // Advances in Imaging and Electron Physics. 2003. V. 128. P. 206–309.
11. Ivanovic I.D. Geometrical description of quantal state determination // Journal of Physics A: Mathematical and General. 1981. dec. V. 14. № 12. P. 3241–3245. Access mode: https://doi.org/10.1088/0305-4470/14/12/019
12. *Englert Berthold-Georg*, *Aharonov Yakir*. The mean king’s problem: prime degrees of freedom // Physics Letters A. 2001. V. 284. № 1. P. 1–5.
13. A New Proof for the Existence of Mutually Unbiased Bases / Bandyopadhyay Somshubhro, Boykin P Oscar, Roychowdhury Vwani, and Vatan Farrokh // Algorithmica. 2002. V. 34. № 4. P. 512–528.
14. *Jagannathan Ramaswamy*. On generalized Clifford algebras and their physical applications // The legacy of Alladi Ramakrishnan in the mathematical sciences. Springer, 2010. P. 465–489.
15. *Кириллов А.А.* Элементы теории представлений. Москва: “Наука”, 1978. С. 343.
16. *Banks T.* Finite Deformations of Quantum Mechanics // arXiv: High Energy Physics – Theory. 2020. Jan. P. 2001.07662.
17. *Kornyak Vladimir V.* Quantum mereology in finite quantum mechanics // Discrete and Continuous Models and Applied Computational Science. 2021. V. 29. № 4. P. 347–360. Access mode: https://journals.rudn.ru/miph/article/view/29428.
18. *Kornyak Vladimir V.* Subsystems of a Closed Quantum System in Finite Quantum Mechanics // Journal of Mathematical Sciences. 2022. Apr. V. 261. P. 717–729.

Access mode:
https://doi.org/10.1007/s10958-022-05783-2

19. *Kornyak Vladimir V.* Decomposition of a Finite Quantum System into Subsystems: Symbolic–Numerical Approach // Programming and Computer Software. 2022. Jul. V. 48. P. 293–300. Access mode: https://doi.org/10.1134/S0361768822020062
20. *Brierley Stephen, Weigert Stefan, Bengtsson Ingemar*. All mutually unbiased bases in dimensions two to five // Quantum Inf. Comput. 2010. V. 10. P. 803–820.
21. *Brierley Stephen, Weigert Stefan*. Constructing mutually unbiased bases in dimension six // Phys. Rev. A. 2009. May. V. 79. P. 052316. Access mode: https://link.aps.org/doi/10.1103/PhysRevA.79.052316.
22. Three numerical approaches to find mutually unbiased bases using Bell inequalities / Colomer Maria Prat, Mortimer Luke, Frérot Iréenée, Farkas Máté, and Acín Antonio // arXiv preprint arXiv:2203.09429. 2022.
23. *Klappenecker Andreas, Rötteler Martin*. Constructions of mutually unbiased bases // International Conference on Finite Fields and Applications / Springer. 2003. P. 137–144.
24. *Bron Coenraad, Kerbosch Joep*. Algorithm 457: finding all cliques of an undirected graph // Communications of The ACM. 1973. V. 16. P. 575–577.
25. *Рейнгольд Э., Нивергельт Ю., Део Н.* Комбинаторные алгоритмы: теория и практика. Москва: “МИР”, 1980. С. 478.
26. *Tarjan Robert Endre*, *Trojanowski Anthony E*. Finding a Maximum Independent Set // SIAM Journal on Computing. 1977. V. 6. № 3. P. 537–546. https://doi.org/10.1137/0206038
27. *Carraghan Randy, Pardalos Panos M*. An exact algorithm for the maximum clique problem // Operations Research Letters. 1990. V. 9. № 6. P. 375–382. Access mode: https://www.sciencedirect.com/science/article/pii/016763779090057C.
28. *Robson John Michael*. Algorithms for Maximum Independent Sets // J. Algorithms. 1986. V. 7. P. 425–440.
29. *Östergård Patric R.J*. A fast algorithm for the maximum clique problem // Discrete Applied Mathematics. 2002. V. 120. № 1. P. 197–207. Special Issue devoted to the 6th Twente Workshop on Graphs and Combinatorial Optimization. Access mode: https://www.sciencedirect.com/science/article/pii/S0166218X01002906.

Complementarity in Finite Quantum Mechanics and Computer-Aided Computations of Complementary Observables

© 2023 г. V. V. Kornyak

Joint Institute for Nuclear Research, Dubna, Moscow oblast, 141980 Russia

e-mail: vkornyak@gmail.com

Mathematical formulation of Bohr’s complementarity principle leads to the concepts of mutually unbiased bases in Hilbert spaces and complementary quantum observables. In this paper, we consider algebraic structures associated with these concepts and their applications to constructive quantum mechanics. We also briefly discuss some computer-algebraic approaches to the problems under consideration and propose an algorithm for solving one of them.

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 70–78

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 517.9+004.4

РЕЗОНАНСЫ И ПЕРИОДИЧЕСКИЕ ДВИЖЕНИЯ МАШИНЫ АТВУДА С ДВУМЯ КОЛЕБЛЮЩИМИСЯ ГРУЗАМИ

© 2023 г. А. Н. Прокопеня[a,*]

[a] *Варшавский университет естественных наук – SGGW*
02-776 Варшава, ул. Новоурсыновска, 159, Польша
**E-mail: alexander_prokopenya@sggw.edu.pl*

Поступила в редакцию 03.08.2022 г.
После доработки 16.10.2022 г.
Принята к публикации 30.10.2022 г.

Обсуждается проблема построения периодических решений уравнений движения машины Атвуда, в которой оба груза одинаковой массы могут колебаться в вертикальной плоскости. Получены дифференциальные уравнения движения системы и описан алгоритм вычисления их решений, определяющих периодические колебания грузов при условии резонанса частот вида $n\omega_1 = m\omega_2$, где n и m – натуральные числа, в виде степенных рядов по малому параметру. Сравнение полученных результатов с соответствующими численными решениями уравнений движения подтверждает их корректность. Все необходимые вычисления выполняются с помощью системы компьютерной алгебры Wolfram Mathematica.

DOI: 10.31857/S0132347423020140, **EDN:** MGUJCT

1. ВВЕДЕНИЕ

Машина Атвуда [1], в которой один из грузов может колебаться в вертикальной плоскости под действием силы тяжести, представляет собой консервативную механическую систему с двумя степенями свободы, которая была предметом исследования многих работ (см., напр., [2–8]). Следует отметить, что уравнения движения такой системы существенно нелинейны и их общее решение не может быть записано в символьной форме. Численное исследование уравнений движения показывает, что машина Атвуда с одним колеблющимся грузом может совершать различные виды движения, например, квазипериодическое и хаотическое движение (см. [3, 5, 7]). В частности, при небольшой разнице масс грузов система может находиться в состоянии динамического равновесия, когда колеблющийся груз меньшей массы уравновешивает груз большей массы. Естественно в таком случае более тяжелый груз также колеблется около некоторого равновесного положения, но его движение происходит только вдоль вертикальной оси. Соответствующее равновесное состояние системы описывается периодическим решением уравнений движения (см. [9]).

Поскольку колебания груза приводят к возрастанию средней силы натяжения нити (см. [8]), можно ожидать, что рассматриваемая система будет находиться в состоянии динамического равновесия и в том случае, когда оба груза имеют одинаковую массу ($m_1 = m_2$), но второй груз также совершает колебания. В работах [10, 11] показано, что такое состояние системы существует, если оба груза совершают синфазные колебания или колеблются в противофазе с одинаковыми амплитудами и частотами. В таком случае грузы можно рассматривать как два одинаковых маятника, длины которых не изменяются. Состояние динамического равновесия существует и в том случае, когда грузы совершают колебания с одинаковыми амплитудами и частотами со сдвигом по фазе $\pm\pi/2$ (см. [11]). При этом длины маятников колеблются в противофазе около некоторого равновесного значения.

В данной работе обсуждается возможность существования периодических движений машины Атвуда с двумя колеблющимися грузами одинаковой массы в более общем случае, когда грузы колеблются с разными частотами ω_1 и ω_2, удовлетворяющими резонансным соотношениям вида $n\omega_1 = m\omega_2$, где n и m – натуральные числа. Основное внимание уделяется описанию алгоритма символьных вычислений, позволяющего найти приближенные решения уравнений движения в виде степенных рядов по малому параметру. Построение и исследование таких решений сводится к выполнению достаточно стандартных, но весьма громоздких символьных вычислений, которые удобно реализовать с помощью систем компьютерной алгебры (см., напр., [12–16]). Хотя

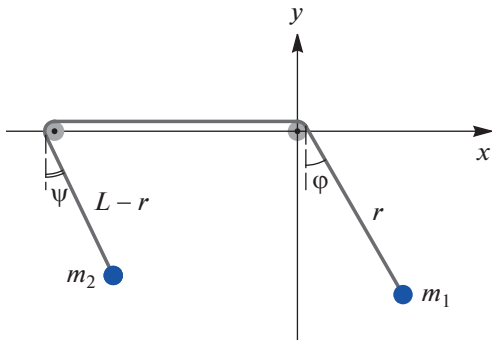

Рис. 1. Машина Атвуда с двумя колеблющимися грузами.

для таких расчетов может быть использована любая доступная система компьютерной алгебры, в данной работе все расчеты и визуализация полученных результатов выполняются с помощью системы *Wolfram Mathematica* [17].

2. ОПИСАНИЕ МОДЕЛИ

Рассматривается машина Атвуда, в которой два груза массами m_1 и m_2 подвешены на нити длиной $(L + b)$, перекинутой через два шкива пренебрежимо малого радиуса, находящихся на расстоянии b друг от друга (рис. 1). Использование двух шкивов не изменяет физической природы классической машины Атвуда (см. [1]) и позволяет избежать столкновений грузов при колебаниях. Геометрическая конфигурация системы определяется углами φ и ψ отклонения грузов m_1 и m_2 от вертикали, а также расстоянием r между грузом m_1 и шкивом. Так как нить предполагается нерастяжимой, расстояние между грузом m_2 и шкивом равняется $(L - r)$.

Система имеет три степени свободы, а ее функция Лагранжа имеет вид (см. [11])

$$\mathcal{L} = \frac{m_1}{2}(\dot r^2 + r^2\dot\varphi^2) + \frac{m_2}{2}(\dot r^2 + (L-r)^2\dot\psi^2) + m_1 g r\cos\varphi + m_2 g(L-r)\cos\psi, \tag{2.1}$$

где точка над символом означает полную производную соответствующей функции по времени, g – ускорение свободного падения. Выражение (2.1) записано в предположении, что радиусы шкивов пренебрежимо малы и изменением длины нити r и $(L - r)$ при колебаниях грузов за счет наматывании нити на шкив можно пренебречь.

Используя функцию Лагранжа (2.1), запишем уравнения движения в виде (см., напр., [18])

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot\varphi}\right) - \frac{\partial\mathcal{L}}{\partial\varphi} = 0, \quad \frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot\psi}\right) - \frac{\partial\mathcal{L}}{\partial\psi} = 0, \quad \frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot r}\right) - \frac{\partial\mathcal{L}}{\partial r} = 0. \tag{2.2}$$

Выполняя в (2.2) дифференцирование с помощью встроенной функции *D* системы *Mathematica* (см. [17]), получаем

$$r\ddot\varphi = -g\sin\varphi - 2\dot r\dot\varphi, \tag{2.3}$$

$$(L-r)\ddot\psi = -g\sin\psi + 2\dot r\dot\psi, \tag{2.4}$$

$$(m_1+m_2)\ddot r = m_1 g\cos\varphi - m_2 g\cos\psi + m_1 r\dot\varphi^2 - m_2(L-r)\dot\psi^2. \tag{2.5}$$

Далее будем предполагать, что массы грузов одинаковы $m_1 = m_2$, и для удобства вычислений введем безразмерные переменные

$$r^*(t^*) = r(t)/R_0, \quad t^* = t\sqrt{g/R_0}, \tag{2.6}$$

где R_0 – длина нити r в положении равновесия $\varphi = 0$, $\psi = 0$ в отсутствие колебаний. В дальнейших вычислениях безразмерные переменные r^*, t^* будем обозначать обычным образом через r, t. Тогда уравнения движения (2.3)–(2.5) принимают вид

$$r\ddot\varphi = -\sin\varphi - 2\dot r\dot\varphi, \tag{2.7}$$

$$(k-r)\ddot\psi = -\sin\psi + 2\dot r\dot\psi, \tag{2.8}$$

$$2\ddot r = \cos\varphi - \cos\psi + r\dot\varphi^2 - (k-r)\dot\psi^2, \tag{2.9}$$

где введен безразмерный параметр $k = L/R_0 > 1$.

В работе [11] показано, что существуют периодические решения уравнений (2.7)–(2.9), которые описывают колебания грузов с одинаковыми частотами и амплитудами со сдвигом по фазе 0, $\pm\pi/2$ или $\pm\pi$. Целью данной работы является рассмотрение более общего случая и построение решений системы (2.7)–(2.9), описывающих состояния динамического равновесия системы, когда оба груза совершают периодические колебания, а их частоты соизмеримы, т.е. удовлетворяют соотношению $n\omega_1 = m\omega_2$, где n и m – натуральные числа. Поскольку взаимодействие двух маятников неизбежно приводит к появлению колебаний с комбинационными частотами $n\omega_1 \pm m\omega_2$ (см., напр., [18]), только в случае соизмеримых частот можно ожидать существования периодических движений системы.

3. ПОСТРОЕНИЕ ПЕРИОДИЧЕСКИХ РЕШЕНИЙ

Отметим, что уравнения (2.7)–(2.9) являются существенно нелинейными, а период колебаний нелинейной системы обычно зависит от амплитуды (см., напр., [18, 19]). С другой стороны, сред-

нее значение силы натяжения нити при колебаниях маятника также зависит от амплитуды (см. [8]) и потому в условиях динамического равновесия системы естественно ожидать, что амплитуды колебаний обоих грузов должны быть одинаковы. Далее будем считать, что амплитуды колебаний грузов определяются параметром ε, который предполагается малым, но конечным. Поскольку при $\varepsilon \to 0$, функции $r(t)$, $\varphi(t)$, $\psi(t)$ должны сводиться к равновесному решению $r = 1$, $\varphi = 0$, $\psi = 0$, для удобства вычислений произведем замену переменных

$$r(t) \to 1 + \varepsilon r(t),$$
$$\varphi(t) \to \sqrt{\varepsilon}\varphi(t), \quad \psi(t) \to \sqrt{\varepsilon}\psi(t), \tag{3.1}$$

и будем предполагать, что $r(t)$, $\varphi(t)$, $\psi(t)$ являются ограниченными осциллирующими функциями. Подставляя (3.1) в (2.7)–(2.9) и заменяя тригонометрические функции их разложениями в степенные ряды с точностью до седьмого порядка включительно, перепишем уравнения движения в виде, удобном для применения теории возмущений (см. [18, 20]):

$$\ddot{\varphi} + \varphi = -\varepsilon\left(r\ddot{\varphi} + 2\dot{r}\dot{\varphi} - \frac{1}{6}\varphi^3\right) - \frac{\varepsilon^2}{120}\varphi^5 + \frac{\varepsilon^3}{5040}\varphi^7,$$
$$(k-1)\ddot{\psi} + \psi = \varepsilon\left(r\ddot{\psi} + 2\dot{r}\dot{\psi} + \frac{1}{6}\psi^3\right) - \frac{\varepsilon^2}{120}\psi^5 + \frac{\varepsilon^3}{5040}\psi^7, \tag{3.2}$$
$$\ddot{r} = -\frac{1}{4}(\varphi^2 - \psi^2) + \frac{1}{2}\dot{\varphi}^2 - \frac{1}{2}(k-1)\dot{\psi}^2 + \frac{\varepsilon}{2}\left(r(\dot{\varphi}^2 + \dot{\psi}^2) + \frac{1}{24}(\varphi^4 - \psi^4)\right) - \frac{\varepsilon^2}{1440}(\varphi^6 - \psi^6).$$

Поскольку предполагается, что частоты колебаний грузов соизмеримы и зависят от амплитуд колебаний, которые в условиях динамического равновесия являются одинаковыми и определяются параметром ε, частоты колебаний можно представить в виде

$$\omega_1 = \omega, \quad \omega_2 = \frac{\omega}{\sqrt{k-1}}, \tag{3.3}$$

где

$$\omega = 1 + \varepsilon\omega_{10} + \varepsilon^2\omega_{20} + \ldots. \tag{3.4}$$

Из (3.3), (3.4) получаем частоты $\omega_1 = 1$, $\omega_2 = 1/\sqrt{k-1}$ гармонических колебаний маятников, определяемых уравнениями (3.2) в случае $\varepsilon = 0$. При подстановке частот (3.3) в уравнение $n\omega_1 = m\omega_2$, где n и m – натуральные числа, находим выражение для начальной длины второго маятника $k = 1 + m^2/n^2$. Так как начальная длина первого маятника равна 1, выбор такой длины второго маятника обеспечивает выполнение условия соизмеримости частот $n\omega_1 = m\omega_2$. Отметим, что неизвестные коэффициенты ω_{10}, ω_{20}, ... в разложении (3.4) далее определяются из условия существования периодических решений.

Для упрощения вычислений введем новую независимую переменную $\tau = \omega t$ и заменим производные по времени в (3.2) согласно правилу

$$\frac{d}{dt} \to \omega\frac{d}{d\tau}, \quad \frac{d^2}{dt^2} \to \omega^2\frac{d^2}{d\tau^2}. \tag{3.5}$$

Решение системы (3.2) будем искать в виде степенных рядов по малому параметру ε:

$$\varphi(\tau) = \varphi_0(\tau) + \varepsilon\varphi_1(\tau) + \varepsilon^2\varphi_2(\tau) + \ldots,$$
$$\psi(\tau) = \psi_0(\tau) + \varepsilon\psi_1(\tau) + \varepsilon^2\psi_2(\tau) + \ldots, \tag{3.6}$$
$$r(\tau) = r_0(\tau) + \varepsilon r_1(\tau) + \varepsilon^2 r_2(\tau) + \ldots.$$

Подставляя (3.4)–(3.6) в (3.2), получаем следующую систему уравнений:

$$\ddot{\varphi}_0 + \varphi_0 + \varepsilon(\ddot{\varphi}_1 + \varphi_1) + \varepsilon^2(\ddot{\varphi}_2 + \varphi_2) + \ldots =$$
$$= -\varepsilon\left(r_0\ddot{\varphi}_0 + 2\dot{r}_0\dot{\varphi}_0 + 2\omega_{10}\ddot{\varphi}_0 - \frac{1}{6}\varphi_0^3\right) -$$
$$- \varepsilon^2\left(\frac{1}{120}\varphi^5 + r_0\ddot{\varphi}_1 + 2\dot{r}_0\dot{\varphi}_1 + (2\omega_{10}r_0 + r_1)\ddot{\varphi}_0 + 2\omega_{10}\ddot{\varphi}_1 + (\omega_{10}^2 + 2\omega_{20})\ddot{\varphi}_0 + 2(\omega_{10}\dot{r}_0 + \dot{r}_1)\dot{\varphi}_0 - \frac{1}{2}\varphi_0^2\varphi_1\right) -$$
$$- \varepsilon^3\left(-\frac{1}{5040}\varphi_0^7 + r_0\ddot{\varphi}_2 + 2\omega_{10}\ddot{\varphi}_2 + (\omega_{10}^2 + 2\omega_{20} + 2\omega_{10}r_0 + r_1)\ddot{\varphi}_1 + 2\dot{r}_2\dot{\varphi}_0 + 2(\omega_{10}\dot{\varphi}_0 + \dot{\varphi}_1)\dot{r}_1 + 2(\omega_{20}\dot{\varphi}_0 + \omega_{10}\dot{\varphi}_1 + \dot{\varphi}_2)\dot{r}_0 + (2(\omega_{10}\omega_{20} + \omega_{30}) + (\omega_{10}^2 + 2\omega_{20})r_0 + 2\omega_{10}r_1 + r_2)\ddot{\varphi}_0 + \frac{1}{24}\varphi_0^4\varphi_1 - \frac{1}{2}\varphi_0(\varphi_1^2 + \varphi_0\varphi_2)\right) + \ldots, \tag{3.7}$$

$$(k-1)\ddot{\psi}_0 + \psi_0 + \varepsilon((k-1)\ddot{\psi}_1 + \psi_1) + \ldots =$$
$$= \varepsilon\left(r_0\ddot{\psi}_0 + 2\dot{r}_0\dot{\psi}_0 - 2(k-1)\omega_{10}\ddot{\psi}_0 + \frac{1}{6}\psi_0^3\right) -$$
$$- \varepsilon^2\left(\frac{1}{120}\psi^5 + (2\omega_{10}(k-1) - r_0)\ddot{\psi}_1 - (2\omega_{10}r_0 + r_1 - (k-1)(\omega_{10}^2 + 2\omega_{20}))\ddot{\psi}_0 - 2\dot{r}_0(\omega_{10}\dot{\psi}_0 + \dot{\psi}_1) - 2\dot{r}_1\dot{\psi}_0 - \frac{1}{2}\psi_0^2\psi_1\right) - \varepsilon^3\left(-\frac{1}{5040}\psi_0^7 + (2\omega_{10}(k-1) - r_0)\ddot{\psi}_2 + ((k-1)(\omega_{10}^2 + 2\omega_{20}) - 2\omega_{10}r_0 - r_1)\ddot{\psi}_1 - 2\dot{r}_2\dot{\psi}_0 - 2(\omega_{10}\dot{\psi}_0 + \dot{\psi}_1)\dot{r}_1 - 2(\omega_{20}\dot{\psi}_0 + \omega_{10}\dot{\psi}_1 + \dot{\psi}_2)\dot{r}_0 + (2(k-1)\times \tag{3.8}$$

$$\times(\omega_{10}\omega_{20}+\omega_{30})-(\omega_{10}^2+2\omega_{20})r_0)\ddot{\psi}_0-(2\omega_{10}\dot{r}_1+$$
$$+r_2)\ddot{\psi}_0+\frac{1}{24}\psi_0^4\psi_1-\frac{1}{2}\psi_0(\psi_1^2+\psi_0\psi_2)\Big)+\ldots,$$
$$\ddot{r}_0+\varepsilon\ddot{r}_1+\varepsilon^2\ddot{r}_2+\ldots=-\frac{1}{4}\Big(\varphi_0^2-\psi_0^2\Big)+\frac{1}{2}\dot{\varphi}_0^2-$$
$$-\frac{1}{2}(k-1)\dot{\psi}_0^2+\varepsilon\Big(\Big(\frac{1}{2}r_0+\omega_{10}\Big)\dot{\varphi}_0^2+$$
$$+\Big(\frac{1}{2}r_0-(k-1)\omega_{10}\Big)\dot{\psi}_0^2+\dot{\varphi}_0\dot{\varphi}_1-(k-1)\dot{\psi}_0\dot{\psi}_1-$$
$$-\frac{1}{2}(\varphi_0\varphi_1-\psi_0\psi_1)+\frac{1}{48}\Big(\varphi_0^4-\psi_0^4\Big)\Big)+$$
$$+\varepsilon^2\Big(\frac{1}{2}r_1(\dot{\varphi}_0^2+\dot{\psi}_0^2)+r_0\Big(\omega_{10}(\dot{\varphi}_0^2+\dot{\psi}_0^2)+\dot{\varphi}_0\dot{\varphi}_1+$$
$$+\dot{\psi}_0\dot{\psi}_1\Big)+2\omega_{10}\dot{\varphi}_0\dot{\varphi}_1-2(k-1)\omega_{10}\dot{\psi}_0\dot{\psi}_1+ \qquad (3.9)$$
$$+\frac{1}{2}\dot{\varphi}_1^2+\dot{\varphi}_0\dot{\varphi}_2-(k-1)\dot{\psi}_0\dot{\psi}_2-\frac{1}{2}(k-1)\dot{\psi}_1^2+$$
$$+\frac{1}{12}(\varphi_0^3\varphi_1-\psi_0^3\psi_1)-\frac{1}{4}(\varphi_1^2-\psi_1^2)-\frac{1}{2}(\varphi_0\varphi_2-$$
$$-\psi_0\psi_2)-\frac{1}{1440}(\varphi_0^6-\psi_0^6)+$$
$$+\frac{1}{2}(\omega_{10}^2+2\omega_{20})\Big(\dot{\varphi}_0^2-(k-1)\dot{\psi}_0^2\Big)\Big)+\ldots.$$

Отметим, что уравнения (3.7)–(3.9) записаны в форме, удобной для вычисления неизвестных функций $r_j(\tau),\varphi_j(\tau),\psi_j(\tau),(j=0,1,2,...)$ в разложениях (3.6). Приравнивая коэффициенты при одинаковых степенях параметра ε в левой и правой части каждого уравнения (3.7)–(3.9), получим систему дифференциальных уравнений, которые можно решать последовательно, выбирая начальные условия таким образом, чтобы получаемые решения определяли периодические движения тел. Соответствующий алгоритм символьных вычислений и используемые функции системы компьютерной алгебры *Wolfram Mathematica* [17] подробно описаны в работе [11]. Поэтому в данной работе сосредоточимся только на основных особенностях вычислений в рассматриваемом случае различных частот колебаний грузов и обсудим получаемые результаты.

Будем считать, что в начальный момент времени обоим грузам, находящимся в положении равновесия $\varphi(0)=0$, $\psi(0)=0$, сообщают некоторые начальные скорости в горизонтальном направлении. Поскольку амплитуды колебаний грузов определяются параметром ε и зависят от их начальных скоростей, без ограничения общности рассуждений будем считать, что начальная скорость первого груза равна $\dot{\varphi}(0)=1$, а начальная скорость второго груза определяется в процессе вычислений из условия существования периодических движений системы. Начальная скорость в радиальном направлении также предполагается равной нулю $\dot{r}(0)=0$, а начальное значение длины $r(0)$ определяется из условия, что частоты колебаний грузов при $\varepsilon=0$ равняются $\omega_1=1$, $\omega_2=1/\sqrt{k-1}=n\omega_1/m$.

В нулевом порядке по ε из уравнений (3.7)–(3.9) получаем

$$\ddot{\varphi}_0+\varphi_0=0, \qquad (3.10)$$
$$(k-1)\ddot{\psi}_0+\psi_0=0, \qquad (3.11)$$
$$\ddot{r}_0=\frac{1}{4}\Big(\psi_0^2-\varphi_0^2\Big)+\frac{1}{2}\dot{\varphi}_0^2-\frac{1}{2}(k-1)\dot{\psi}_0^2. \qquad (3.12)$$

Уравнения (3.10), (3.11) определяют гармонические колебания маятников и их решения, удовлетворяющие начальным условиям $\varphi_0(0)=0$, $\dot{\varphi}_0(0)=1$, $\psi_0(0)=0$, можно представить в виде

$$\varphi_0(\tau)=\sin(\tau),\quad \psi_0(\tau)=\sin\Big(\frac{\tau}{\sqrt{k-1}}\Big). \qquad (3.13)$$

Амплитуды колебаний в (3.13) одинаковы, что обеспечивает равенство нулю постоянной составляющей функции в правой части уравнения (3.12) и не приводит к появлению квадратичной зависимости от времени функции $r_0(\tau)$, которая является решением уравнения (3.9) при $\varepsilon=0$. Напомним, что интересующее нас решение (3.6) описывает малые колебания длины $r(\tau)$ около некоторого равновесного значения, а амплитуды колебаний переменных $\varphi(\tau)$ и $\psi(\tau)$ определяются параметром ε (см. (3.1)). Поэтому далее предполагаем, что функции $\varphi_j(\tau)$, $\psi_j(\tau)$, $r_j(\tau)$, $j=1,2,\ldots$ удовлетворяют начальным условиям $\varphi_j(0)=0$, $\dot{\varphi}_j(0)=0$, $\psi_j(0)=0$, $\dot{r}_j(0)=0$, и будем искать такие решения уравнений (3.7)–(3.9), которые описывают малые колебания функций $\varphi_j(\tau)$, $\psi_j(\tau)$, $r_j(\tau)$.

Подставляя решения (3.13) в (3.12), получаем дифференциальное уравнение

$$\ddot{r}_0=\frac{3}{8}\cos(2\tau)-\frac{3}{8}\cos\Big(\frac{2\tau}{\sqrt{k-1}}\Big). \qquad (3.14)$$

Для получения общего решения уравнения (3.14) достаточно дважды проинтегрировать его правую часть с помощью встроенной функции *Integrate*[#, τ, τ]&. В результате находим

$$r_0=-\frac{3}{32}\cos(2\tau)+\frac{3(k-1)}{32}\cos\Big(\frac{2\tau}{\sqrt{k-1}}\Big)+r_{00}, \qquad (3.15)$$

где r_{00} – произвольная постоянная. Отметим, что вторая постоянная, возникающая при интегрировании уравнения второго порядка (3.14), полагается равной нулю, так как в противном случае искомое решение (3.15) будет содержать линейно растущую функцию времени и не будет периодическим.

Далее выполняем подобные вычисления в первом порядке по ε. Из уравнений (3.7)–(3.9) получаем

$$\ddot{\varphi}_1 + \varphi_1 = -r_0\ddot{\varphi}_0 - 2\dot{r}_0\dot{\varphi}_0 - 2\omega_{10}\ddot{\varphi}_0 + \frac{1}{6}\varphi_0^3, \quad (3.16)$$

$$(k-1)\ddot{\psi}_1 + \psi_1 = r_0\ddot{\psi}_0 + 2\dot{r}_0\dot{\psi}_0 - 2(k-1)\omega_{10}\ddot{\psi}_0 + \frac{1}{6}\psi_0^3, \quad (3.17)$$

$$\ddot{r}_1 = \left(\frac{1}{2}r_0 + \omega_{10}\right)\dot{\varphi}_0^2 + \left(\frac{1}{2}r_0 - (k-1)\omega_{10}\right)\dot{\psi}_0^2 + \dot{\varphi}_0\dot{\varphi}_1 - (k-1)\dot{\psi}_0\dot{\psi}_1 - \frac{1}{2}(\varphi_0\varphi_1 - \psi_0\psi_1) + \frac{1}{48}(\varphi_0^4 - \psi_0^4). \quad (3.18)$$

Подставляя решения (3.13), (3.15) в уравнение (3.16) и преобразуя его правую часть в линейную комбинацию тригонометрических функций с помощью функции *TrigReduce*, получаем

$$\ddot{\varphi}_1 + \varphi_1 = \left(2\omega_{10} + r_{00} - \frac{1}{64}\right)\sin(\tau) - \frac{53}{192}\sin(3\tau) + \left(\frac{3(k-1)}{64} - \frac{3\sqrt{k-1}}{16}\right)\sin\left(\tau - \frac{2\tau}{\sqrt{k-1}}\right) + \left(\frac{3(k-1)}{64} + \frac{3\sqrt{k-1}}{16}\right)\sin\left(\tau + \frac{2\tau}{\sqrt{k-1}}\right). \quad (3.19)$$

Аналогичным образом приводим уравнение (3.17) к виду

$$(k-1)\ddot{\psi}_1 + \psi_1 = \left(2\omega_{10} - \frac{r_{00}}{k-1} - \frac{1}{64}\right)\sin\left(\frac{\tau}{\sqrt{k-1}}\right) - \left(\frac{3}{64(k-1)} - \frac{3}{16\sqrt{k-1}}\right)\sin\left(2\tau - \frac{\tau}{\sqrt{k-1}}\right) + \left(\frac{3}{64(k-1)} + \frac{3}{16\sqrt{k-1}}\right)\sin\left(2\tau + \frac{\tau}{\sqrt{k-1}}\right) - \frac{53}{192}\sin\left(\frac{3\tau}{\sqrt{k-1}}\right). \quad (3.20)$$

Дифференциальные уравнения (3.19), (3.20) описывают вынужденные колебания переменных $\varphi_1(\tau)$, $\psi_1(\tau)$ и их решения будут ограниченными осциллирующими функциями только при условии, что правые части этих уравнений не содержат резонансных членов, пропорциональных $\cos(\tau)$, $\sin(\tau)$ и $\cos\left(\frac{\tau}{\sqrt{k-1}}\right)$, $\sin\left(\frac{\tau}{\sqrt{k-1}}\right)$ соответственно, которые приводят к неограниченному возрастанию амплитуды колебаний (см. [18, 19]). Поскольку при произвольном значении параметра k вычисления довольно громоздки, далее в качестве примера рассмотрим случай $k = 5/4$ или $2\omega_1 = \omega_2$. Подставляя значение $k = 5/4$ в уравнения (3.19), (3.20) и приравнивая к нулю коэффициенты при $\sin(\tau)$ и $\sin(2\tau)$, получаем систему двух уравнений

$$2\omega_{10} + r_{00} + \frac{7}{128} = 0, \quad 2\omega_{10} - 4r_{00} - \frac{19}{64} = 0. \quad (3.21)$$

Решая систему (3.21), находим значения неизвестных параметров

$$r_{00} = -\frac{9}{128}, \quad \omega_{10} = \frac{1}{128}. \quad (3.22)$$

Подставляя параметры (3.22) и $k = 5/4$ в уравнение (3.19), с помощью встроенной функции *DSolve* находим решение, удовлетворяющее начальным условиям $\varphi_1(0) = \dot{\varphi}_1(0) = 0$

$$\varphi_1(\tau) = \frac{1}{6144}\left(-312\sin(\tau) + 149\sin(3\tau) - 27\sin(5\tau)\right). \quad (3.23)$$

Соответствующее решение уравнения (3.20), удовлетворяющее начальному условию $\psi_1(0) = 0$, имеет вид

$$\psi_1(\tau) = C_1\sin(2\tau) - \frac{3}{16}\sin(4\tau) + \frac{53}{1536}\sin(6\tau), \quad (3.24)$$

где C_1 – произвольная постоянная, которая далее будет найдена из условия периодичности движения системы.

Подставляя решения (3.13), (3.15), (3.23), (3.24) в уравнение (3.18) и преобразуя его правую часть в линейную комбинацию тригонометрических функций с учетом (3.22), получаем дифференциальное уравнение для определения функции $r_1(\tau)$

$$\ddot{r}_1 = -\frac{13}{1024} - \frac{1}{4}C_1 - \frac{285}{8192}\cos(2\tau) + \frac{33}{2048}\cos(4\tau) - \frac{3}{4}C_1\cos(4\tau) + \frac{1461}{8192}\cos(6\tau) - \frac{105}{2048}\cos(8\tau). \quad (3.25)$$

При условии $C_1 = -13/256$ постоянная составляющая функции в правой части (3.25), которая привела бы к квадратичной зависимости функции $r_1(\tau)$ от времени, обнуляется, что приводит к осциллирующему частному решению

$$r_1(t) = r_{10} + \frac{285}{32768}\cos(2\tau) - \frac{111}{32768}\cos(4\tau) - \frac{487}{98304}\cos(6\tau) + \frac{105}{131072}\cos(8\tau), \quad (3.26)$$

где r_{10} – произвольная постоянная, которая далее будет найдена из условия периодичности функций $\varphi_2(\tau)$, $\psi_2(\tau)$.

Действительно, приравнивая коэффициенты при ε^2 в левой и правой части каждого уравнения (3.7), (3.8) и подставляя в них найденные решения (3.13), (3.15), (3.22), (3.23), (3.24), (3.26), получим следующие дифференциальные уравнения:

$$\ddot{\varphi}_2 + \varphi_2 = \left(2\omega_{20} + r_{10} - \frac{15}{131072}\right)\sin(\tau) + \frac{1179}{32768}\sin(3\tau) - \frac{37931}{491520}\sin(5\tau) - \frac{11003}{1572864}\sin(7\tau) + \frac{1815}{524288}\sin(9\tau), \tag{3.27}$$

$$\frac{1}{4}\ddot{\psi}_2 + \psi_2 = \left(2\omega_{20} - 4r_{10} + \frac{75}{32768}\right)\sin(2\tau) - \frac{793}{12288}\sin(4\tau) - \frac{14841}{65536}\sin(6\tau) + \frac{14515}{49152}\sin(8\tau) - \frac{17439}{327680}\sin(10\tau). \tag{3.28}$$

Условие отсутствия резонансных членов в правых частях уравнений (3.27), (3.28), пропорциональных $\sin(\tau)$ и $\sin(2\tau)$ соответственно, дает

$$r_{10} = \frac{63}{131072}, \quad \omega_{20} = -\frac{3}{16384}. \tag{3.29}$$

С учетом (3.29), решение уравнения (3.27), удовлетворяющее начальным условиям $\varphi_2(0) = \dot{\varphi}_2(0) = 0$, получается в виде

$$\varphi_2(\tau) = -\frac{40459}{12582912}\sin(\tau) - \frac{1179}{262144}\sin(3\tau) + \frac{37931}{11796480}\sin(5\tau) + \frac{11003}{75497472}\sin(7\tau) - \frac{363}{8388608}\sin(9\tau). \tag{3.30}$$

Решение уравнения (3.28), удовлетворяющее начальному условию $\psi_2(0) = 0$, получаем в виде

$$\psi_2(\tau) = C_2\sin(2\tau) + \frac{793}{36864}\sin(4\tau) + \frac{14841}{524288}\sin(6\tau) - \frac{2903}{147456}\sin(8\tau) + \frac{5813}{2621440}\sin(10\tau), \tag{3.31}$$

где C_2 – постоянная, которую определим из условия существования периодического решения $r_2(\tau)$. Приравнивая коэффициенты при ε^2 в левой и правой части уравнения (3.9) и подставляя в них найденные решения (3.13), (3.15), (3.22)–(3.24), (3.26), (3.29)–(3.31), получаем дифференциальное уравнение, определяющее $r_2(\tau)$, в виде

$$\ddot{r}_2 = \frac{5581}{1572864} - \frac{1}{4}C_2 - \frac{165409}{25165824}\cos(2\tau) + \left(\frac{161629}{6291456} - \frac{3}{4}C_2\right)\cos(4\tau) - \frac{386787}{16777216}\cos(6\tau) - \frac{342979}{6291456}\cos(8\tau) + \frac{2325035}{50331648}\cos(10\tau) - \frac{6795}{1048576}\cos(12\tau). \tag{3.32}$$

При условии $C_2 = 5581/393216$ постоянная составляющая функции в правой части (3.32) обнуляется, а его интегрирование приводит к осциллирующему частному решению

$$r_2(t) = r_{20} + \frac{165409}{100663296}\cos(2\tau) - \frac{94657}{100663296}\cos(4\tau) + \frac{128929}{201326592}\cos(6\tau) + \frac{342979}{402653184}\cos(8\tau) - \frac{465007}{1006632960}\cos(10\tau) + \frac{755}{16777216}\cos(12\tau), \tag{3.33}$$

где r_{20} – постоянная, которая далее будет найдена из условия периодичности функций $\varphi_3(\tau)$, $\psi_3(\tau)$.

Описанный процесс последовательного решения дифференциальных уравнений, которые получаются путем приравнивания коэффициентов при одинаковых степенях параметра ε в левой и правой части каждого из уравнений (3.7)–(3.9), можно продолжить и вычислить решения (3.6) и частоты (3.3) с требуемой точностью, хотя в более высоких порядках такие вычисления становятся все более громоздкими. Например, с точностью до третьего порядка по ε находим

$$\varphi(\tau) = \sqrt{\varepsilon}\sin(\tau) + \frac{\varepsilon^{3/2}}{6144}(-312\sin(\tau) + 149\sin(3\tau) - 27\sin(5\tau)) + \varepsilon^{5/2}\left(-\frac{40459}{12582912}\sin(\tau) - \frac{1179}{262144}\sin(3\tau) + \frac{37931}{11796480}\sin(5\tau) + \frac{11003}{75497472}\sin(7\tau) - \frac{363}{8388608}\sin(9\tau)\right) + \varepsilon^{7/2}\left(\frac{6414499\sin(\tau)}{16106127360} + \frac{103864093\sin(3\tau)}{154618822656} - \frac{23354665}{51539607552}\sin(5\tau) - \frac{1310507}{90194313216}\sin(7\tau) - \right. \tag{3.34}$$

$$-\frac{26414627}{966367641600}\sin(9\tau)+\frac{8712713}{429496729600}\sin(11\tau)-$$
$$-\frac{92069}{51539607552}\sin(13\tau)\Big),$$

$$\psi(\tau)=\sqrt{\varepsilon}\sin(2\tau)+$$
$$+\frac{\varepsilon^{3/2}}{1536}\big(-78\sin(2\tau)-288\sin(4\tau)+$$
$$+53\sin(6\tau)\big)+\varepsilon^{5/2}\Big(\frac{5581}{393216}\sin(2\tau)+$$
$$+\frac{793}{36864}\sin(4\tau)+\frac{14841}{524288}\sin(6\tau)-$$
$$-\frac{2903}{147456}\sin(8\tau)+\frac{5813}{2621440}\sin(10\tau)\Big)+ \qquad (3.35)$$
$$+\varepsilon^{7/2}\Big(\frac{2351739\sin(2\tau)}{2684354560}-\frac{871399\sin(4\tau)}{100663296}-$$
$$-\frac{5770387}{1610612736}\sin(6\tau)-\frac{624487}{314572800}\sin(8\tau)+$$
$$+\frac{10923889}{1610612736}\sin(10\tau)-\frac{15973393}{7549747200}\sin(12\tau)+$$
$$+\frac{961939}{5637144576}\sin(14\tau)\Big),$$

$$r(\tau)=1+\frac{\varepsilon}{128}\big(-9+24\sin^4\tau\big)+\varepsilon^2\Big(\frac{63}{131072}+$$
$$+\frac{285}{32768}\cos(2\tau)-\frac{111}{32768}\cos(4\tau)-$$
$$-\frac{487}{98304}\cos(6\tau)+\frac{105}{131072}\cos(8\tau)\Big)+$$
$$+\varepsilon^3\Big(\frac{4869}{16777216}+\frac{165409}{100663296}\cos(2\tau)- \qquad (3.36)$$
$$-\frac{94657}{100663296}\cos(4\tau)+$$
$$+\frac{128929}{201326592}\cos(6\tau)+\frac{342979}{402653184}\cos(8\tau)-$$
$$-\frac{465007}{1006632960}\cos(10\tau)+\frac{755}{16777216}\cos(12\tau)\Big).$$

При этом частоты колебаний грузов определяются выражениями (3.3), причем в случае $k = 5/4$ имеем $\omega_1 = \omega$, $\omega_2 = 2\omega$, где

$$\omega=1+\frac{\varepsilon}{128}-\frac{3\varepsilon^2}{16384}-\frac{125219\varepsilon^3}{100663296}. \qquad (3.37)$$

Описанные вычисления можно продолжить и получить функции (3.6) с необходимой точностью, хотя в высших порядках по ε вычисления становятся все более громозкими и для их выполнения требуется применение систем компьютерной алгебры. При использовании системы *Wolfram Mathematica*, например, большинство операций можно реализовать, используя встроенные функции такие как *Expand*, *TrigExpand*, *Collect*, *Coefficient*, *D*, *Integrate*, *Series*, *Normal*, *DSolve*.

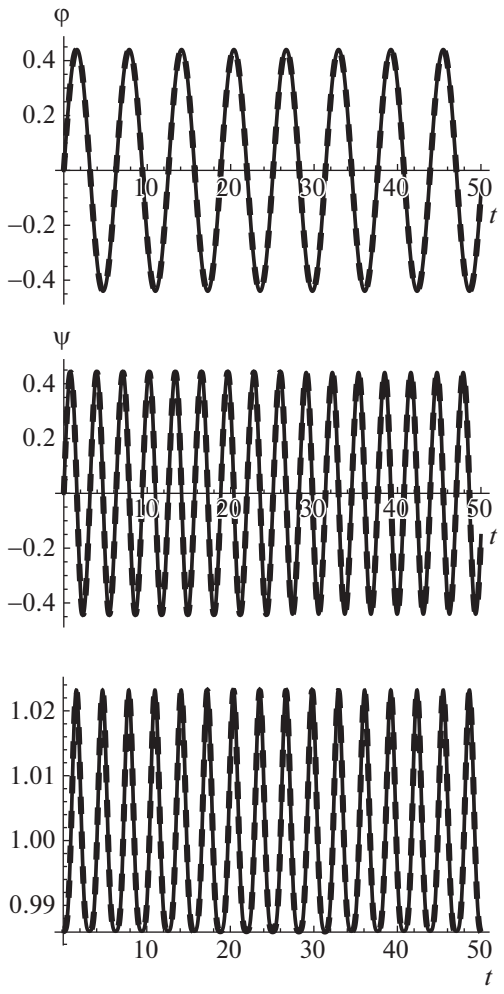

Рис. 2. Сравнение аналитического и численного решений системы (2.7)–(2.9) при $\varepsilon = 0.2$, $k = 5/4$.

Используя найденные решения (3.33)–(3.35), можно вычислить начальные значения функций $\varphi(0)$, $\psi(0)$, $r(0)$ и их производных $\dot\varphi(0)$, $\dot\psi(0)$, $\dot r(0)$ при выбранно значении параметра ε, а затем найти соответствующие численные решения уравнений движения (2.7)–(2.9) с помощью встроенной функции *NDSolve*. Визуализация найденных аналитических решений при $\varepsilon = 0.2$ (сплошные тонкие линии на рис. 2) демонстрирует хорошее совпадение с численными решениями (штриховые линии на рис. 2).

4. ЗАКЛЮЧЕНИЕ

В настоящей работе обсуждается проблема построения периодических решений уравнений движения машины Атвуда с двумя колеблющимися грузами одинаковой массы, которые описывают состояние динамического равновесия системы. Существование такого равновесного состояния не является очевидным и, как показывают вычисления, возможно только при таком выборе начальных условий, когда частоты колебаний грузов ω_1, ω_2 являются соизмеримыми, т.е. удовлетворяют соотношению $n\omega_1 = m\omega_2$, где n и m – натуральные числа. Поскольку фактически рассматриваемая машина Атвуда представляет собой систему из двух маятников переменной длины, их взаимодействие неизбежно приводит к появлению колебаний с комбинационными частотами $n\omega_1 \pm m\omega_2$, а такое движение может быть периодическим только в случае соизмеримых частот. Соответствующие периодические решения уравнений движения найдены в виде степенных рядов по малому параметру ε, который определяет амплитуды колебаний. Последовательно описаны символьные вычисления, необходимые для определения коэффициентов этих рядов, а сами функции найдены с точностью до третьего порядка по ε включительно. Сравнение найденного аналитического решения с численным решением уравнений движения показало справедливость полученных теоретических результатов. Следует отметить, что реализовать описанные символьные вычисления удается только выбирая конкретные натуральные числа n и m в выражении $n\omega_1 = m\omega_2$, поскольку в общем случае трудно выделить резонансные члены в уравнениях движения в каждом порядке по ε.

Отметим также, что в данной работе все вычисления и визуализация результатов выполнены с использованием системы компьютерной алгебры *Wolfram Mathematica*.

СПИСОК ЛИТЕРАТУРЫ

1. *Atwood G.* A Treatisa on the Rectilinear Motion and Rotation of Bodies. Cambridge University Press, 1784.
2. *Tufillaro N.B., Abbott T.A., Griffiths D.J.* Swinging Atwood's machine // American Journal of Physics. 1984. V. 52(3.1). P. 895–903.
3. *Tufillaro N.B.* Motions of a swinging Atwood's machine // J. Physique. 1985. V. 46. P. 1495–1500.
4. *Tufillaro N.B.* Integrable motion of a swinging Atwood's machine // Amer. J. Phys. 1986. V. 54. P. 142–143.
5. *Casasayas J., Nunes T.A., Tufillaro N.B.* Swinging Atwood's machine: integrability and dynamics // J. Physique. 1990. V. 51. P. 1693–1702.
6. *Yehia H.M.* On the integrability of the motion of a heavy particle on a tilted cone and the swinging Atwood's machine // Mech. R. Comm. 2006. V. 33(2.5). P. 711–716.
7. *Pujol O., Pérez J.P., Ramis J.P., Simo C., Simon S., Weil J.A.* Swinging Atwood machine: Experimental and numerical results, and a theoretical study // Physica D. 2010. V. 239(3.3). P. 1067–1081.
8. *Prokopenya A.N.* Motion of a swinging Atwood's machine: simulation and analysis with Mathematica // Mathematics in Computer Science. 2017. V. 11(3–4). P. 417–425.
9. *Прокопеня А.Н.* Построение периодического решения уравнений движения обобщенной машины Атвуда с применением компьютерной алгебры // Программирование. 2020. Т. 46(2.2). С. 53–59.
10. *Prokopenya A.N.* Modelling Atwood's Machine with Three Degrees of Freedom // Mathematics in Computer Science. 2019. V. 13(1–2). P. 247–257.
11. *Прокопеня А.Н.* Поиск равновесных состояний машины Атвуда с двумя колеблющимися грузами с применением компьютерной алгебры // Программирование. 2021. Т. 47(2.1). С. 56–64.
12. *Абрамов С.А., Зима Е.Б., Ростовцев В.А.* Компьютерная алгебра // Программирование. 1992. № 5. С. 4–25.
13. *Васильев Н.Н., Еднерал В.Ф.* Компьютерная алгебра в физических и математических приложениях // Программирование. 1994. № 1. С. 70–82.
14. *Прокопеня А.Н.* Некоторые алгоритмы символьных вычислений в исследованиях проблем космической динамики // Программирование. 2006. Т. 32(2.2). С. 16–22.
15. *Прокопеня А.Н.* Символьные вычисления в исследованиях устойчивости решений линейных систем дифференциальных уравнений с периодическими коэффициентами // Программирование. 2007. Т. 33(2.2). С. 9–16.
16. *Прокопеня А.Н.* Нормализация гамильтониана в ограниченной задаче многих тел методами компьютерной алгебры // Программирование. 2012. Т. 38(2.3). С. 65–78.
17. *Wolfram S.* An elementary introduction to the Wolfram Language, 2nd ed. Champaign, IL, USA, Wolfram Media, 2017.
18. *Ландау Л.Д., Лифшиц Е.М.* Механика. 4-е изд. М.: Наука, 1988, 216 с.
19. *Маркеев А.П.* Теоретическая механика. Ижевск: НИЦ "Регулярная и хаотическая динамика", 2001. 592 с.
20. *Nayfeh A.H.* Introduction to Perturbation Techniques. New York: John Wiley & Sons, 1981, 519 p.

Resonances and Periodic Motions of Atwood's Machine with Two Oscillating Weights

© 2023 г. A. N. Prokopenya

Warsaw University of Life Sciences, Warsaw, 02-776 Poland

e-mail: alexander_prokopenya@sggw.edu.pl

The problem of constructing periodic solutions to the equations of motion of Atwood's machin in which both weights have the same mass and can oscillate in the vertical plane is discussed. Differential equations governing the motion of this system are derived, and an algorithm for calculating their solutions that determine periodic oscillations of the weights under the condition that the oscillation frequencies are in resonance $n\omega_1 = m\omega_2$, where n and m are natural numbers, is proposed. These solutions are obtained in the form of series in a small parameter. The comparison of the results with numerical solutions of the equations of motion confirm the validity of the obtained solutions. All computations are performed using the computer algebra system Wolfram Mathematica.

ПРОГРАММИРОВАНИЕ, 2023, № 5, с. 79–86

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 004.421.6+519.61

ЭФФЕКТИВНЫЕ НИЖНИЕ ГРАНИЦЫ ДЛЯ РАНГА МАТРИЦЫ И ПРИЛОЖЕНИЯ

© 2023 г. О. А. Зверков[a],*, А. В. Селиверстов[a],**

[a] *Институт проблем передачи информации им. А.А. Харкевича Российской академии наук*
127051 Москва, Большой Каретный пер., д. 19, Россия

**E-mail: zverkov@iitp.ru*

***E-mail: slvstv@iitp.ru*

Поступила в редакцию 26.06.2022 г.
После доработки 27.07.2022 г.
Принята к публикации 30.10.2022 г.

Предложена эффективно проверяемая нижняя граница для ранга разреженной вполне неразложимой квадратной матрицы, содержащей по два ненулевых элемента в каждой строке и каждом столбце. Ранг такой матрицы равен порядку или отличается на единицу. Построены базисы специального вида в пространствах квадратичных форм от фиксированного числа переменных. Существование таких базисов позволило нам обосновать эвристический алгоритм для решения задачи распознавания, проходит ли данное аффинное подпространство через вершину многомерного единичного куба. В худшем случае этот алгоритм может вернуть уведомление о неопределенности результата вычисления, но для общего подпространства достаточно малой размерности корректно отвергает вход. Алгоритм реализован на языке Python. В ходе тестирования получены оценки времени работы этой реализации алгоритма.

DOI: 10.31857/S0132347423020176, **EDN:** MHEYDZ

1. ВВЕДЕНИЕ

Python – современный язык программирования общего назначения, вышедший в последние годы на первое место в различных рейтингах популярности и особенно востребованный в научных приложениях. Сильными сторонами Python считаются легкость начального освоения, быстрота процесса разработки, лаконичность и удобочитаемость текстов программ. В качестве основного недостатка обычно называется относительно низкая производительность программ, обусловленная в основном интерпретируемой природой языка и динамической типизацией. Однако этот недостаток во многом компенсируется доступностью большого разнообразия библиотек, предоставляющих Python-разработчикам доступ к эффективным реализациям различных алгоритмов, а также возможностью подключения собственных программных модулей, написанных на компилируемых языках [1]. В контексте нашей задачи удобной особенностью Python является встроенная в язык поддержка работы с целыми числами неограниченной битовой длины, а также с рациональными числами.

Ранг $n \times n$ матрицы над полем можно вычислить, используя полиномиальное число процессоров и выполнив на каждом из них лишь $O(\log_2^2 n)$ операций над этим полем [2, 3]. Эффективное распараллеливание возможно не только для вычисления ранга, но также для вычисления обратной матрицы, LDU-разложения квадратной матрицы и разложения Холецкого симметричной положительно определенной матрицы [4, 5]. С другой стороны, сложность вычисления ранга [6] и характеристического многочлена [7–9] близка к сложности матричного умножения. Для циркулянтов известны легко проверяемые условия невырожденности [10]. Теоретический интерес представляет задача проверки линейной независимости системы многочленов от нескольких переменных. Иногда такие многочлены могут быть заданы короткими выражениями или арифметическими схемами, хотя матрицы коэффициентов таких многочленов имеют большой размер. Поэтому важны легко проверяемые оценки ранга матрицы. Обычно вычисления выполняются над полем рациональных чисел, но символьные вычисления можно проводить и над полями алгебраических чисел [11, 12].

Точки детерминантального многообразия соответствуют (с точностью до умножения на ненулевое число) матрицам фиксированного размера, ранг которых ограничен сверху. Детерминантальное многообразие может иметь очень большую степень [13, 14]. Поэтому над алгебраически за-

мкнутым полем нельзя проверить значение ранга матрицы посредством неветвящейся программы небольшого размера. С другой стороны, на этих многообразиях лежат линейные подпространства, что иногда можно использовать для оценки ранга [15].

Используя оценки ранга, мы рассмотрим эвристический алгоритм проверки инцидентности данного подпространства и какой-либо вершины единичного куба. Такая проверка эквивалентна поиску (0, 1)-решения для системы уравнений

$$\begin{cases} x_0 = 1 \\ x_j = \ell_j(x_0, \ldots, x_s), \quad j > s \\ x_i \in \{0,1\}, \quad i > 0, \end{cases}$$

где через ℓ_j обозначены линейные формы [16].

Хотя уже известны эвристические алгоритмы для решения этой задачи при дополнительных ограничениях [17–19], принято считать, что эта задача трудная для вычисления в худшем случае. Поэтому актуально создание новых алгоритмов, у которых вычислительная сложность в среднем ограничена многочленом от длины входа при других ограничениях.

Говоря о задачах распознавания, мы предполагаем три варианта ответа: вход может быть не только принят или отвергнут, но также возможно явное уведомление о неопределенности выбора. При этом ответ должен быть получен за конечное время и без ошибок, а при выполнении легко проверяемого условия уведомление о неопределенности может быть выдано лишь на малой доле входов среди всех входов данной длины [20, 21]. При оценке алгебраической сложности достаточно требовать, чтобы уведомление о неопределенности выдавалось на множестве входов, на которых обращается в нуль некоторый многочлен, отличный от тождественно нулевого [16, 22].

В разделе 2 представлены теоретические результаты. В разделе 3 описана реализация алгоритма на языке Python. В разделе 4 дано краткое заключение.

2. ТЕОРЕТИЧЕСКИЕ РЕЗУЛЬТАТЫ

Элементами (0, 1)-матрицы служат лишь нули или единицы. Подматрицей называется матрица, полученная удалением некоторых строк и столбцов. Матрица размера $n \times n$ называется вполне неразложимой, когда в ней нет нулевой подматрицы размера $s \times (n - s)$ ни для какого s.

Рассмотрим квадратные матрицы, имеющие в каждой строке и каждом столбце по два ненулевых элемента. Если такого типа (0, 1)-матрица вполне неразложимая, то ее перманент равен двум [23]. В этом случае значение определителя принадлежит множеству $\{-2, 0, 2\}$. Очевидно, вырождена любая такая (0, 1)-матрица над полем характеристики два. Примером вырожденной матрицы над произвольным полем служит 2×2 матрица ранга один, каждый элемент которой равен единице. Другим примером служит (0, 1)-циркулянт

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Эта матрица тоже вырожденная, но ее ранг равен трем. Характеристический многочлен этой матрицы равен $x^4 - 4x^3 + 6x^2 - 4x$. Мы дадим нижнюю оценку ранга таких матриц.

Теорема 1. *Дана вполне неразложимая $n \times n$ матрица A, где в каждой строке и каждом столбце ровно по два ненулевых элемента. Ранг матрицы A ограничен снизу:* $\operatorname{rank}(A) \geq n - 1$.

Доказательство. Условие полной неразложимости не нарушается при перестановке строк или столбцов. Поэтому без ограничения общности полагаем, что на главной диагонали матрицы A все элементы ненулевые. Тогда матрица A равна сумме двух невырожденных матриц $A = D + CP$, где обе матрицы C и D диагональные и невырожденные, а через P обозначена матрица перестановки с нулями на главной диагонали. Поскольку матрица A вполне неразложимая, перестановка P состоит из одного цикла длины n.

Для любой матрицы перестановки Q матрица QAQ^{-1} получается из матрицы A согласованной перестановкой строк и столбцов. При этом элементы на главной диагонали остаются на главной диагонали. Поэтому для некоторой матрицы перестановки Q в матрице QAQ^{-1} отличны от нуля элементы на главной диагонали, непосредственно над главной диагональю и еще один элемент в первом столбце и последней строке. Такая матрица QAQ^{-1} имеет вид (для $n > 6$)

$$\begin{pmatrix} * & * & 0 & \cdots & 0 & 0 & 0 \\ 0 & * & * & \cdots & 0 & 0 & 0 \\ 0 & 0 & & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & * & * & 0 \\ 0 & 0 & 0 & \cdots & 0 & * & * \\ * & 0 & 0 & \cdots & 0 & 0 & * \end{pmatrix},$$

где символом * отмечены ненулевые элементы матрицы. Поскольку при перестановке строк или столбцов ранг матрицы не меняется, ранг матрицы QAQ^{-1} совпадает с рангом матрицы A. Для за-

вершения доказательства теоремы достаточно найти невырожденную $(n-1)\times(n-1)$ подматрицу B в матрице QAQ^{-1}. Выберем подматрицу B, удаляя из матрицы A первую строку и первый столбец. Эта матрица верхнетреугольная. Каждый элемент на главной диагонали матрицы B отличен от нуля. Поэтому матрица B невырожденная. □

В теореме 1 условие полной неразложимости важно. Например, для четного n у блочно-диагональной матрицы размера $n\times n$, на диагонали которой стоят 2×2 блоки по четыре единицы в каждом, ранг равен $n/2$.

Далее мы рассматриваем поля характеристики нуль. Рассмотрим задачу об инцидентности данного аффинного подпространства и какой-либо вершины n-мерного куба. Метод состоит в попытке построить алгебраическую гиперповерхность малой степени, проходящую через каждую вершину куба, но не пересекающую данное аффинное подпространство. Вычислительная сложность этого метода зависит от степени гиперповерхности. С другой стороны, этот метод эффективен лишь при условии, что размерность аффинного подпространства достаточно мала. Пусть это подпространство определяется системой из m линейных уравнений. Размерность линейного пространства форм степени d от переменных $x_0, ..., x_n$ равна биномиальному коэффициенту $\binom{n+d}{d}$. Поэтому рассматриваемый метод эффективен в среднем, когда число уравнений удовлетворяет неравенству типа $m \ge n - \sqrt{d(d-1)n - o(n)}$, где через d обозначена степень искомой гиперповерхности. Мы ограничимся случаем $d = 2$, когда гиперповерхностью служит квадрика [16].

Пусть n-мерный куб вложен в проективное пространство, а однородные координаты вершин принадлежат множеству $\{0,1\}$, но ни одна из вершин куба не лежит на бесконечно удаленной гиперплоскости $x_0 = 0$. Известно [24], что квадрика, проходящая через каждую вершину этого куба, определяется квадратичной формой типа

$$\lambda_1 x_1(x_1 - x_0) + \cdots + \lambda_n x_n(x_n - x_0).$$

Если подпространство задано системой уравнений $x_j = \ell_j(x_0,\ldots,x_{n-m})$, где $n-m < j \le n$, то существование квадрики, проходящей через каждую вершину куба и пересекающей это подпространство лишь на бесконечности, эквивалентно разрешимости уравнения

$$\sum_{k=1}^{n-m}\lambda_k x_k(x_k - x_0) + \sum_{j=n-m+1}^{n}\lambda_j \ell_j(\ell_j - x_0) = x_0^2$$

относительно переменных $\lambda_1, ..., \lambda_n$. В свою очередь, эта задача эквивалентна разрешимости системы линейных уравнений. По теореме Кронекера–Капелли, эта проверка сводится к сравнению рангов двух матриц. Алгоритм 1 не обеспечивает полиномиальную вычислительную сложность в худшем случае. Но иногда он гораздо эффективнее полного перебора вершин куба.

В алгоритме 1 первые $n-m$ столбцов матрицы A всегда содержат по два ненулевых элемента 1 и -1. В столбце с номером $j > n-m$ элементы равны многочленам второй степени от коэффициентов линейной формы ℓ_j.

Алгоритм 1. Проверка существования вершины куба, инцидентной данному подпространству.

Input: целые числа $0 < m < n$ и линейные формы $\ell_j(x_0,\ldots,x_{n-m})$, где $n-m < j \le n$.

1: Элементами матрицы A служат коэффициенты формы

$$\sum_{k=1}^{n-m}\lambda_k x_k(x_k - x_0) + \sum_{j=n-m+1}^{n}\lambda_j \ell_j(\ell_j - x_0),$$

где строки матрицы A соответствуют мономам второй степени от переменных x_0, ..., x_{n-m}, а столбцы – переменным $\lambda_1, ..., \lambda_n$;

2: Расширенная матрица B получается из матрицы A добавлением столбца, в котором единица стоит в строке, соответствующей моному x_0^2, а остальные элементы этого столбца равны нулю;

3: **if** $\operatorname{rank}(A) = \operatorname{rank}(B)$ **then** вход отвергается

4: **else** уведомление о неопределенности.

Рассмотрим пример, соответствующий прямой на плоскости. Пусть $n = 2$, $m = 1$, линейная форма $\ell_2 = x_0 + x_1$. Получаем уравнение

$$-\lambda_1 x_0 x_1 + \lambda_1 x_1^2 + \lambda_2 x_0 x_1 + \lambda_2 x_1^2 = x_0^2.$$

Здесь три монома от переменных x_0 и x_1. Это x_0^2, x_0x_1 и x_1^2. Соответствующая матрица A содержит два столбца и три строки:

$$A = \begin{pmatrix} 0 & 0 \\ -1 & 1 \\ 1 & 1 \end{pmatrix}.$$

Расширенная матрица равна

$$B = \begin{pmatrix} 0 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix}.$$

Здесь $\mathrm{rank}(A) = 2$ и $\mathrm{rank}(B) = 3$. Поэтому искомые λ_1 и λ_2 не существуют. Ответ алгоритма 1 состоит в уведомлении о неопределенности.

Рассмотрим другой пример, соответствующий прямой в пространстве. Пусть $n = 3$, $m = 2$, линейные формы $\ell_2 = 2x_0 + x_1$ и $\ell_3 = 3x_1$. Получаем уравнение $\lambda_1(-x_0x_1 + x_1^2) + \lambda_2(2x_0^2 + 3x_0x_1 + x_1^2) + \lambda_3(-3x_0x_1 + 9x_1^2) = x_0^2$. Здесь снова три монома от переменных x_0 и x_1. Это x_0^2, x_0x_1 и x_1^2. Но матрица A содержит три столбца и три строки:

$$A = \begin{pmatrix} 0 & 2 & 0 \\ -1 & 3 & -1 \\ 1 & 1 & 9 \end{pmatrix}.$$

Расширенная матрица равна

$$B = \begin{pmatrix} 0 & 2 & 0 & 1 \\ -1 & 3 & -3 & 0 \\ 1 & 1 & 9 & 0 \end{pmatrix}.$$

Матрица A невырожденная, ранги матриц A и B совпадают. Алгоритм 1 отвергает вход.

Следующая теорема обеспечивает достаточное условие типа $m \ge n - \sqrt{2n - o(n)}$, при котором алгоритм 1 отвергает большую долю входов при фиксированных значениях n и m. Доказательство основано на построении базиса специального вида в пространстве квадратичных форм.

Теорема 2. *Даны положительные целые числа n и $m < n$, для которых выполнено неравенство $2n \ge (n - m + 1)(n - m + 2)$. Для почти каждого набора из m линейных форм $\ell_j(x_0, \ldots, x_{n-m})$, где $n - m < j \le n$, найдутся значения коэффициентов $\lambda_1, \ldots, \lambda_n$, для которых выполнено равенство квадратичных форм*

$$\sum_{k=1}^{n-m} \lambda_k x_k (x_k - x_0) + \sum_{j=n-m+1}^{n} \lambda_j \ell_j (\ell_j - x_0) = x_0^2.$$

Если для некоторого $\varepsilon > 0$ коэффициенты линейных форм ℓ_j независимо и равномерно распределены на множестве из $\lceil 2n/\varepsilon \rceil$ чисел, то вероятность отсутствия искомых значений $\lambda_1, \ldots, \lambda_n$ не превышает ε.

Доказательство. Рассмотрим матрицу A из алгоритма 1. По условию теоремы, в матрице A число строк $r = (n - m + 1)(n - m + 2)/2$ не превышает числа столбцов n. В этом случае достаточным условием существования искомых $\lambda_1, \ldots, \lambda_n$ служит равенство $\mathrm{rank}(A) = r$. Каждый элемент матрицы A равен либо константе, либо многочлену второй степени от коэффициентов линейных форм ℓ_j. Следовательно, минор порядка r матрицы A равен многочлену от коэффициентов линейных форм ℓ_j, степень которого не превышает $2r$.

По лемме Шварца–Зиппеля [25], если этот многочлен не равен нулю тождественно, то вероятность обращения в нуль не превышает $2r/\lceil 2n/\varepsilon \rceil \le \varepsilon$.

Осталось показать, что для некоторого набора форм ℓ_j угловой минор порядка r матрицы A отличен от нуля. При этом достаточно рассмотреть набор форм, коэффициенты которых не обязательно принадлежат указанному в условии множеству. Положим $\ell_{n-m+k} = x_0 + x_k$ для $1 \le k \le n - m$. Обозначим через $f(i,k)$ номер пары индексов $1 \le i < k \le n - m$, принимающий значения от 1 до $(n - m)(n - m - 1)/2$. Положим $\ell_j = x_i + x_k$ для $j = 2(n - m) + f(i,k)$. При $j = r$ полагаем $\ell_r = 2x_0$.

Для индексов $1 \le i < k \le n - m$ моному x_ix_k соответствует строка, где отличен от нуля лишь элемент в столбце $j = 2(n - m) + f(i,k)$. Этот элемент равен двум. С другой стороны, в столбцах с номерами $1 \le j \le n - m$ по два ненулевых элемента: -1 в строке, соответствующей моному x_0x_j, и 1 в строке, соответствующей моному x_j^2. Переставим строки в матрице A в соответствии с таким порядком мономов: $x_0x_1, \ldots, x_0x_{n-m}, x_1^2, x_2^2, \ldots, x_{n-m}^2$, мономы x_ix_k для $1 \le i < k \le n - m$ и последний x_0^2. Угловая подматрица порядка r примет вид

$$\begin{pmatrix} -I & I & * & 0 \\ I & I & * & 0 \\ 0 & 0 & 2I' & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix},$$

где через I обозначена единичная матрица порядка $n - m$, через I' – единичная матрица порядка $(n - m)(n - m - 1)/2$, а через 0 – нулевая матрица. Эта подматрица невырожденная, поскольку элементарные преобразования строк дают треугольную матрицу с ненулевыми элементами на главной диагонали

$$\begin{pmatrix} -I & I & * & 0 \\ 0 & 2I & * & 0 \\ 0 & 0 & 2I' & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}.$$

Следовательно, $\mathrm{rank}(A) = r$. □

Заметим, что в доказательстве теоремы 2 используется семейство разреженных матриц. Поэтому разреженность матрицы A не служит препятствием для успешного применения обсуждаемого алгоритма. Более того, для разреженных матриц меньше сложность вычисления ранга, что

позволяет работать в больших размерностях. В доказательстве теоремы 2 рассмотрен частный случай, когда матрица A имеет полный ранг. Поэтому вычисление ранга можно заменить на проверку невырожденности матрицы A. Однако это было бы неоправданным ограничением применимости алгоритма 1.

3. РЕАЛИЗАЦИЯ

Обозначим через L матрицу, составленную из коэффициентов линейных форм ℓ_j в алгоритме 1 и теореме 2. Первая строка матрицы L соответствует форме ℓ_{n-m+1}, а последняя строка – форме ℓ_n. Например, для двух линейных форм $\ell_2 = 2x_0 + x_1$ и $\ell_3 = 3x_1$ эта матрица равна

$$L = \begin{pmatrix} 2 & 1 \\ 0 & 3 \end{pmatrix}.$$

Сопоставление матрице L матрицы B требует меньше времени, чем вычисление рангов матриц A и B. Такое преобразование реализует функция `compose_b()`, которая получает на вход матрицу L и возвращает матрицу B, где обе матрицы представлены массивами NumPy. Порядок строк в B соответствует лексикографическому мономиальному упорядочению, а первая строка матрицы – моному x_0^2. Матрица A легко получается удалением последнего столбца из матрицы B. Функция `compose_b()` реализована на языке Python с использованием библиотеки NumPy.

При использовании целых чисел фиксированной битовой длины, NumPy позволяет значительно повысить эффективность работы с матрицами (как по скорости, так и по памяти) по сравнению со встроенными типами данных языка Python, например списками чисел [26]. При работе с другими типами данных, включая целые неограниченной длины и рациональные числа, эти преимущества в основном утрачиваются. Однако в любом случае NumPy позволяет кратко записывать стандартные операции над матрицами и их частями, например, как в нашем случае, выделение подматрицы, поэлементное сложение и умножение строк и т. п.

Через $s = n - m$ обозначена размерность подпространства, для которого проверяется инцидентность вершине n-мерного куба. Через `height` обозначено число мономов второй степени от переменных $x_0, \ldots, x_s$, равное $(s+1)(s+2)/2$.

Для оценки времени работы функции `compose_b()` при n = `height` генерировалась $(n-s)\times(s+1)$ матрица L, элементами которой служат случайно выбранные целые числа из отрезка $[-10^9, 10^9]$. Время, затраченное на вычисление матрицы B, показано в таблице 1. В третьем столбце таблицы 1 указаны результаты для 64-битовых целых чисел, а в последнем столбце – для целых чисел неограниченной битовой длины, вычисления над которыми реализованы в языке Python. Мы использовали версии Python 3.10.4 и NumPy 1.22.4. Вычисления проведены на персональном компьютере на базе процессора Intel® Core i5-3570 и с оперативной памятью 16 гигабайтов. Поэтому для больших матриц вычисления с числами неограниченной битовой длины не были проведены из-за переполнения оперативной памяти, хотя для 64-битовых чисел вычисления продолжены вплоть до значений $s = 300$.

Одновременное вычисление рангов матриц A и B по данной на вход матрице B также реализовано на языке Python. Для вычислений с рациональными числами используется класс Fraction из стандартной библиотеки Python.

Вычисление ранга основано на приведении матрицы к ступенчатому виду. Поскольку матрица A служит подматрицей в B, для обеих матриц A и B такое преобразование можно сделать одновременно. Это позволяет примерно вдвое уменьшить время работы по сравнению с независимым вычислением рангов двух матриц.

Чтобы оценить время для вычисления функции `rank_ab()`, при n = `height` генерировались $(n-s)\times(s+1)$ матрицы L, элементами которых служат случайно выбранные целые числа из отрезка $[-N, N]$. Время вычисления для разных значений границы диапазона $N \in \{10^3, 10^6, 10^9\}$ показано в таблице 2. Чем больше размер матрицы, тем быстрее возрастает время работы при увеличении средней битовой длины элементов матрицы.

Отметим, что реализованный в библиотеке NumPy метод вычисления ранга, основанный на сингулярном разложении матрицы, в этой задаче применять небезопасно из-за возможных ошибок, связанных с использованием чисел с плавающей запятой. Например, при значениях $k \geq 16$ вычисляемый в NumPy 1.22.4 таким неточным способом ранг диагональной 2×2 матрицы с элементами 1 и 10^k на главной диагонали оказывается равным единице. Поэтому для вычисления ранга мы использовали новую реализацию известного алгоритма Гаусса приведения матрицы к ступенчатому виду, которая позволяет работать с числами неограниченной битовой длины.

В итоге реализацией алгоритма 1 служит функция `may_have_01solution()`, которая получает на вход матрицу L. Значение равно `True`, когда алгоритм 1 дал бы неопределенный ответ. Значение `False` означает, что алгоритм 1 отвергает вход. Входом может служить матрица с рациональными элементами.

Таблица 1. Время в секундах для вычисления матрицы B при различных значениях s и n для двух типов целых чисел: 64-битовых и неограниченной битовой длины

Параметры		Тип целых чисел	
s	n	64-битовые	без ограничений
20	231	0.01	0.02
30	496	0.03	0.08
40	861	0.07	0.20
50	1326	0.14	0.50
60	1891	0.25	1.00
70	2556	0.43	1.90
80	3321	0.68	3.20
90	4186	1.10	5.00
100	5151	1.50	8.20
110	6216	2.20	12
120	7381	3.00	16
130	8646	4.20	24
140	10011	5.40	36
150	11476	7.50	51
160	13041	9.50	66
170	14706	13	89
180	16471	15	
190	18336	20	
200	20301	23	
210	22366	29	
220	24531	34	
230	26796	43	
240	29161	50	
250	31626	62	
260	34191	83	
270	36856	89	
280	39621	100	
290	42486	130	
300	45451	170	

Листинги обсуждаемых функций и примеры их использования доступны по адресу http://lab6.iitp.ru/-/qualg

4. ЗАКЛЮЧЕНИЕ

Реализован на языке Python эвристический алгоритм полиномиального времени для распознавания подпространств достаточно малой размерности, инцидентных какой-либо из вершин единичного многомерного куба, хотя в худшем случае эта задача считается вычислительно трудной. Алгоритм либо корректно отвергает вход, либо сообщает об отсутствии очевидного препятствия к инцидентности. В частности, полученные ограничения на сложность этой задачи могут быть полезны для оценки надежности криптографических протоколов, основанных на поиске (0, 1)-решения системы уравнений [19, 27].

Таблица 2. Время в секундах для вычисления ранга матрицы B при различных значениях параметров s, n и N

Параметры		N		
s	n	10^3	10^6	10^9
5	21	0.02	0.02	0.02
6	28	0.04	0.05	0.07
7	36	0.09	0.10	0.19
8	45	0.21	0.34	0.50
9	55	0.45	0.77	1.20
10	66	0.91	1.70	2.70
11	78	1.80	3.50	5.70
12	91	3.30	6.90	12
13	105	6	13	23
14	120	11	24	42
15	136	18	42	76
16	153	30	73	130
17	171	50	120	230
18	190	79	200	380
19	210	120	330	610
20	231	190	510	960

Алгоритм использует вычисление ранга матрицы. Полученные в теореме 1 оценки ранга неразложимой разреженной матрицы позволяют, в частности, тестировать вычисление ранга матрицы большого порядка.

Авторы благодарны анонимному рецензенту за сделанные замечания, позволившие улучшить статью.

СПИСОК ЛИТЕРАТУРЫ

1. *Геворкян М.Н., Королькова А.В., Кулябов Д.С., Севастьянов Л.А.* Пример модульного расширения системы компьютерной алгебры // Программирование. 2020. № 2. С. 30–37. DOI: Перевод: Programming and Computer Software. 2020. V. 46. № 2. P. 98–104. https://doi.org/10.31857/S0132347420020065
2. *Chistov A.L.* Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic // In: *L. Budach* (eds) Fundamentals of Computation Theory. FCT 1985. Lecture Notes in Computer Science, vol. 199. Springer, Berlin, Heidelberg, 1985. P. 63–69. https://doi.org/10.1007/BFb0028792
3. *Mulmuley K.* A fast parallel algorithm to compute the rank of a matrix over an arbitrary field // Combinatori-

ca. 1987. V. 7. № 1. P. 101–104.
https://doi.org/10.1007/BF02579205

4. *Malaschonok G., Tchaikovsky I.* About big matrix inversion // In: *Abramov S.A., Sevastyanov L.A.* (eds) Computer algebra. Moscow: MAKS Press, 2021. P. 81–84.
https://doi.org/10.29003/m2019.978-5-317-06623-9

5. *Малашонок Г.И., Сидько А.А.* Суперкомпьютерная среда выполнения для рекурсивных матричных алгоритмов // Программирование. 2022. № 2. С. 33–46. DOI: Перевод: Programming and Computer Software. 2022. V. 48. P. 90–101.
https://doi.org/10.31857/S0132347422020091

6. *Cheung H.Y., Kwok T.C., Lau L.C.* Fast matrix rank algorithms and applications // Journal of the ACM. 2013. V. 60. № 5. Article № 31. P. 1–25.
https://doi.org/10.1145/2528404

7. *Abdeljaoued J., Malaschonok G.I.* Efficient algorithms for computing the characteristic polynomial in a domain // Journal of Pure and Applied Algebra. 2001. V. 156. P. 127–145.
https://doi.org/10.1016/S0022-4049(99)00158-9

8. *Переславцева О.Н.* О вычислении характеристического полинома матрицы // Дискретная математика. 2011. Т. 23. № 1. С. 28–45. DOI: Перевод: Discrete Mathematics and Applications. 2011. V. 21. № 1. P. 109–128.
https://doi.org/10.4213/dm1128

9. *Neiger V., Pernet C.* Deterministic computation of the characteristic polynomial in the time of matrix multiplication // Journal of Complexity. 2021. V. 67. № 101572. P. 1–35.
https://doi.org/10.1016/j.jco.2021.101572

10. *Chen Z.* On nonsingularity of circulant matrices // Linear Algebra and its Applications. 2021. V. 612. P. 162–176.
https://doi.org/10.1016/j.laa.2020.12.010

11. *Алаев П.Е., Селиванов В.Л.* Поля алгебраических чисел, вычислимые за полиномиальное время. I // Алгебра и логика. 2019. Т. 58. № 6. С. 673–705. DOI: Перевод: Algebra Logic. 2020. V. 58. P. 447–469.
https://doi.org/10.33048/alglog.2019.58.601

12. *Алаев П.Е., Селиванов В.Л.* Поля алгебраических чисел, вычислимые за полиномиальное время. II // Алгебра и логика. 2021. Т. 60. № 6. С. 533–548. DOI: Перевод: Algebra Logic. 2022. V. 60. P. 349–359.
https://doi.org/10.33048/alglog.2021.60.601

13. *Harris J., Tu L.W.* On symmetric and skew-symmetric determinantal varieties // Topology. 1984. V. 23. № 1. P. 71–84.
https://doi.org/10.1016/0040-9383(84)90026-0

14. *Harris J.* Algebraic geometry. Springer-Verlag New York, 1992. 330 p.
https://doi.org/10.1007/978-1-4757-2189-8

15. *Rubei E.* Affine subspaces of matrices with constant rank // Linear Algebra and its Applications. 2022. V. 644. № 1. P. 259–269.
https://doi.org/10.1016/j.laa.2022.03.002

16. *Селиверстов А.В.* Двоичные решения для больших систем линейных уравнений // Прикладная дискретная математика. 2021. № 52. С. 5–15.
https://doi.org/10.17223/20710410/52/1

17. *Кузюрин Н.Н.* Полиномиальный в среднем алгоритм в целочисленном линейном программировании // Сибирский журнал исследования операций. 1994. Т. 1. № 3. С. 38–48.

18. *Kuzyurin N.N.* An integer linear programming algorithm polynomial in the average case // In: *Korshunov A.D.* (eds.) Discrete Analysis and Operations Research. Mathematics and Its Applications. V. 355. P. 143–152. Springer, Dordrecht, 1996.
https://doi.org/10.1007/978-94-009-1606-7

19. *Pan Y., Zhang F.* Solving low-density multiple subset sum problems with SVP oracle // Journal of Systems Science and Complexity. 2016. V. 29. P. 228–242.
https://doi.org/10.1007/s11424-015-3324-9

20. *Рыбалов А.Н.* О генерической сложности проблемы о сумме подмножеств для полугрупп целочисленных матриц // Прикладная дискретная математика. 2020. № 50. С. 118–126.
https://doi.org/10.17223/20710410/50/9

21. *Рыбалов А.Н.* О генерической сложности проблемы вхождения для полугрупп целочисленных матриц // Прикладная дискретная математика. 2022. № 55. С. 95–101.
https://doi.org/10.17223/20710410/55/7

22. *Селиверстов А.В.* Эвристические алгоритмы распознавания некоторых кубических гиперповерхностей // Программирование. 2021. № 1. С. 65–72. DOI: Перевод: Programming and Computer Software. 2021. V. 47. № 1. P. 50–55.
https://doi.org/10.31857/S0132347421010106

23. *Minc H.* (0,1)-matrices with minimal permanents // Israel Journal of Mathematics. 1973. V. 15. P. 27–30.
https://doi.org/10.1007/BF02771770

24. *Seliverstov A.V., Lyubetsky V.A.* About forms equal to zero at each vertex of a cube // Journal of Communications Technology and Electronics. 2012. V. 57. № 8. P. 892–895.
https://doi.org/10.1134/S1064226912080049

25. *Schwartz J.T.* Fast probabilistic algorithms for verification of polynomial identities // Journal of the ACM. 1980. V. 27. № 4. P. 701–717.
https://doi.org/10.1145/322217.322225

26. *Harris C.R., Millman K.J., van der Walt S.J. et al.* Array programming with NumPy // Nature. 2020. V. 585. № 7825. P. 357–362.
https://doi.org/10.1038/s41586-020-2649-2

27. *Chen Y.A., Gao X.S.* Quantum algorithm for Boolean equation solving and quantum algebraic attack on cryptosystems // Journal of Systems Science and Complexity. 2022. V. 35. P. 373–412.
https://doi.org/10.1007/s11424-020-0028-6

Effective Lower Bounds on the Matrix Rank and Their Applications

© 2023 г. O. A. Zverkov[a,#], and A. V. Seliverstov[a,##]

[a] *Institute for Information Transmission Problems (Kharkevich Institute), Russian Academy of Sciences, Bol'shoi Karetnyi per. 19/1, Moscow, 127051 Russia*

[#] *e-mail: zverkov@iitp.ru*

[##] *e-mail: slvstv@iitp.ru*

We propose an efficiently verifiable lower bound on the rank of a sparse fully indecomposable square matrix that contains two non-zero entries in each row and each column. The rank of this matrix is equal to its order or differs from it by one. Bases of a special type are constructed in the spaces of quadratic forms in a fixed number of variables. The existence of these bases allows us to substantiate a heuristic algorithm for recognizing whether a given affine subspace passes through a vertex of a multidimensional unit cube. In the worst case, the algorithm may output a computation denial warning; however, for the general subspace of sufficiently small dimension, it correctly rejects the input. The algorithm is implemented in Python. The running time of its implementation is estimated in the process of testing.