ISSN 1818-1015
eISSN 2313-5417

2023 Volume 30 No 1

MODELING AND ANALYSIS OF INFORMATION SYSTEMS

SCIENTIFIC JOURNAL

Start date of publication — 1999
Published quarterly

FOUNDER
P.G. Demidov Yaroslavl State University

EDITORIAL OFFICE
14 Sovetskaya str., Yaroslavl 150003, Russian Federation

Website: http://mais-journal.ru
E-mail: mais@uniyar.ac.ru
Phone: +7 (4852) 79-77-73

2023 Том 30 № 1

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ

НАУЧНЫЙ ЖУРНАЛ

Издается с 1999 года
Выходит 4 раза в год

УЧРЕДИТЕЛЬ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Ярославский государственный университет им. П. Г. Демидова»

РЕДАКЦИЯ
ул. Советская, 14, Ярославль, 150003, Российская Федерация
Website: http://mais-journal.ru
E-mail: mais@uniyar.ac.ru
Телефон: +7 (4852) 79-77-73

Свидетельство о регистрации СМИ ПИ № ФС 77–66186 от 20.06.2016 выдано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций. Подписной индекс — 31907 в Объединенном каталоге «Пресса России». Технический редактор, компьютерная вёрстка – М. С. Каряева. Корректор английского текста – М. С. Комар. Подписано в печать 16.03.2023. Дата выхода в свет 31.03.2023. Формат 200×265 мм. Объем 100 с. Тираж 32 экз. Свободная цена. Заказ 002/023. Адрес типографии: ул. Советская, 14, оф. 109, Ярославль, 150003, Россия. Адрес издателя: Ярославский государственный университет им. П. Г. Демидова, ул. Советская, 14, Ярославль, 150003, Россия.
Содержание предназначено для детей старше 12 лет.

Contents

Содержание

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 1, 2023
journal homepage: www.mais-journal.ru

ALGORITHMS

The Optimized Algorithm of Finding the Shortest Path in a Multiple Graph

A. V. Smirnov[1]

DOI: 10.18255/1818-1015-2023-1-6-15

[1]P. G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 05C38, 05C65
Research article
Full text in Russian

Received January 18, 2023
After revision February 20, 2023
Accepted February 22, 2023

In this paper, we study undirected multiple graphs of any natural multiplicity $k > 1$. There are edges of three types: ordinary edges, multiple edges and multi-edges. Each edge of the last two types is a union of k linked edges, which connect 2 or $(k + 1)$ vertices, correspondingly. The linked edges should be used simultaneously. If a vertex is incident to a multiple edge, it can be also incident to other multiple edges and it can be the common end of k linked edges of some multi-edge. If a vertex is the common end of some multi-edge, it cannot be the common end of another multi-edge.
As for an ordinary graph, we can define the integer function of the length of an edge for a multiple graph and set the problem of the shortest path joining two vertices. Any multiple path is a union of k ordinary paths, which are adjusted on the linked edges of all multiple and multi-edges. In the article, we optimize the algorithm of finding the shortest path in an arbitrary multiple graph, which was obtained earlier. We show that the optimized algorithm is polynomial. Thus, the problem of the shortest path is polynomial for any multiple graph.

Keywords: multiple graph; multiple path; shortest path; reachability set; polynomial algorithm

INFORMATION ABOUT THE AUTHORS

Alexander Valeryevich Smirnov
correspondence author

orcid.org/0000-0002-0980-2507. E-mail: alexander_sm@mail.ru
PhD, Associate Professor, Department of Theoretical Computer Science.

Funding: This work was supported by P. G. Demidov Yaroslavl State University Project № VIP-016.

For citation: A. V. Smirnov, "The Optimized Algorithm of Finding the Shortest Path in a Multiple Graph", *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 6-15, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 1, 2023
сайт журнала: www.mais-journal.ru

Оптимизированный алгоритм поиска кратчайшего пути в кратном графе

А. В. Смирнов[1]

DOI: 10.18255/1818-1015-2023-1-6-15

[1]Ярославский государственный университет им. П. Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 519.17
Научная статья
Полный текст на русском языке

Получена 18 января 2023 г.
После доработки 20 февраля 2023 г.
Принята к публикации 22 февраля 2023 г.

В статье рассматриваются неориентированные кратные графы произвольной натуральной кратности $k > 1$. Кратный граф содержит ребра трех типов: обычные, кратные и мультиребра. Ребра последних двух типов представляют собой объединение k связанных ребер, которые соединяют 2 или $(k + 1)$ вершину соответственно. Связанные ребра могут использоваться только согласованно. Если вершина инцидентна кратному ребру, то она может быть инцидентна другим кратным ребрам, а также она может быть общим концом k связанных ребер мультиребра. Если вершина является общим концом мультиребра, то она не может быть общим концом никакого другого мультиребра.

Как и для обычного графа, для кратного графа можно ввести целочисленную функцию длины ребра и поставить задачу о кратчайшем пути между двумя вершинами. Кратный путь является объединением k обычных путей, согласованных на связанных ребрах кратных и мультиребер. В статье оптимизирован полученный ранее алгоритм поиска кратчайшего пути в произвольном кратном графе. Показано, что оптимизированный алгоритм полиномиален. Таким образом, задача о кратчайшем пути является полиномиальной для любого кратного графа.

Ключевые слова: кратный граф; кратный путь; кратчайший путь; множество достижимости; полиномиальный алгоритм

ИНФОРМАЦИЯ ОБ АВТОРАХ

Александр Валерьевич Смирнов
автор для корреспонденции

orcid.org/0000-0002-0980-2507. E-mail: alexander_sm@mail.ru
канд. физ.-мат. наук, доцент, кафедра теоретической информатики.

Финансирование: Работа выполнена в рамках инициативной НИР ЯрГУ им. П. Г. Демидова № VIP-016.

Для цитирования: A. V. Smirnov, “The Optimized Algorithm of Finding the Shortest Path in a Multiple Graph”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 6-15, 2023.

Введение

В данной статье мы рассмотрим задачу о *кратчайшем пути* в кратном графе. *Кратные графы* содержат три типа ребер (обычные, кратные и мультиребра) и являются обобщением обычных графов – по сути, обычный граф имеет кратность $k = 1$. Определение кратного графа кратности $k > 1$ было сформулировано в статье [1]. Там же была поставлена задача о кратчайшем кратном пути между двумя вершинами, дано определение связности кратного графа, предложены полиномиальные алгоритмы ее проверки. В отличие от обычного графа, в связном кратном графе путь между двумя вершинами существует не всегда, и в работе [1] обоснован критерий существования кратного пути между двумя вершинами, а также получены быстрые полиномиальные алгоритмы для проверки условия критерия.

В статье [2] получен полиномиальный алгоритм поиска кратчайшего кратного пути в *делимом графе* – частном случае кратного графа. Особенностью делимых графов является возможность представления графа в виде объединения k частей. Каждая часть – это обычный граф, в котором присутствует ровно одно связанное ребро каждого кратного и мультиребра. При этом никакие две части не содержат одинаковых ребер. Кроме того, в работе [2] предложена модификация указанного алгоритма для случая произвольного кратного графа. Однако модифицированный алгоритм является экспоненциальным по параметру k в общем случае.

В данной статье мы оптимизируем общий алгоритм поиска кратчайшего кратного пути. В результате будет получен полиномиальный алгоритм для произвольного кратного графа.

1. Постановка задачи о кратчайшем кратном пути

Напомним несколько понятий, связанных с кратными графами и путями. Поскольку данная статья продолжает исследование, описанное в работе [2], здесь будут сформулированы только самые важные определения. Поясняющие примеры и ряд связанных определений были подробно рассмотрены в статьях [1, 2].

Определение 1. Кратный граф G *произвольной натуральной кратности* $k > 1$ *– это граф, вершины которого могут соединяться ребрами одного из 3 видов:*

1. Обычное ребро e^o*; множество обычных ребер обозначим через* E^o.
2. Кратное ребро e^k *между двумя вершинами, которое состоит из* k *одинаковых связанных ребер; связанные ребра кратного ребра могут использоваться только согласованно; множество кратных ребер обозначим через* E^k.
3. Связанное ребро e *между двумя вершинами, имеющее один общий конец с другим* $(k-1)$ *ребром (у любых двух из* k *связанных ребер только один конец является общим); множество связанных общей вершиной ребер будем называть* мультиребром e^m*; связанные ребра мультиребра могут использоваться только согласованно; множество мультиребер обозначим через* E^m.

Если вершина инцидентна какому-либо кратному ребру, то она может быть инцидентна другим кратным ребрам, а также она может быть общим концом какого-либо мультиребра.

Если вершина является общим концом какого-либо мультиребра, то она не может быть общим концом никакого другого мультиребра.

Если вершина является отдельным концом мультиребра или инцидентна обычному ребру, то она не может быть общим концом мультиребра и не может быть инцидентна кратному ребру.

Множества вершин и ребер графа G обозначим через V и E соответственно. Заметим, что $E = E^o \cup E^k \cup E^m$.

В данной статье рассматриваются только неориентированные кратные графы.

Определение 2. Обычной вершиной *назовем вершину, которая инцидентна обычному ребру или является отдельным концом мультиребра.*

Кратной вершиной *назовем вершину, которая инцидентна кратному ребру или является общим концом мультиребра.*

Из определения 1 следует, что множества обычных и кратных вершин не пересекаются. При этом кратная вершина может быть соединена с обычными только посредством мультиребра.

Определение 3. *$S(x, y) = \cup_{i=1}^{k} S^i(x, y)$ является* кратным путем из вершины x в вершину y *в графе $G(V, E)$, если выполнены следующие условия:*

1. *$S^i(x, y) = \left(\{x, v_1^i\}, \{v_1^i, v_2^i\}, \ldots, \{v_{l_i-1}^i, v_{l_i}^i\}, \{v_{l_i}^i, y\}\right)$, где $l_i \geqslant 0$, – последовательность ребер, представляющая собой обычный (некратный) путь из x в y, где каждое ребро $\{a, b\}$ является либо обычным ребром в графе $G(V, E)$, либо i-ым связанным ребром кратного или мультиребра. Значения l_i и l_j $(i \neq j)$ не согласовываются и могут быть как равными, так и различными. Если в путь $S(x, y)$ не входит ни одного кратного или мультиребра, то $S^2(x, y) = S^3(x, y) = \ldots = S^k(x, y) = \varnothing$.*
2. *Любая обычная вершина может встретиться в $S^i(x, y)$ несколько раз, то есть $S^i(x, y)$ может содержать циклы.*
3. *Никакая кратная вершина не может встретиться в $S^i(x, y)$ дважды.*
4. *Любое обычное ребро может встречаться в $S^i(x, y)$ несколько раз, причем направления, в которых оно проходится в разных вхождениях, могут не совпадать.*
5. *Обычное ребро, входящее в $S^i(x, y)$, может также входить в любой $S^j(x, y)$, $j \neq i$.*
6. *Все пути $S^i(x, y)$ согласованы (одинаковы) на общей части. Это условие означает, что если связанное ребро какого-то кратного или мультиребра входит в некоторый путь $S^i(x, y)$, то остальные связанные ребра должны входить во все $S^j(x, y)$, $j \neq i$ (по одному связанному ребру в каждый $S^j(x, y)$). При этом порядок вхождения всех кратных и мультиребер во все $S^i(x, y)$ одинаков.*

   Фактически это значит, что если e_1 и e_2 – это два ребра пути $S(x, y)$, каждое из которых либо кратное, либо мультиребро, и в проекции $S^i(x, y)$ связанное ребро из e_1 проходится раньше связанного ребра из e_2, то во всех остальных проекциях $S^j(x, y)$ связанные ребра из e_2 могут проходиться только после связанных ребер из e_1.
7. *Если $S(x, y)$ содержит мультиребро $\{x_0, \{x_1, \ldots, x_k\}\}$, проходимое в направлении от общего конца, то он не может содержать никакого другого мультиребра $\{y_0, \{x_1, \ldots, x_k\}\}$, проходимого в том же направлении. Аналогичное условие должно выполняться и в случае движения к общему концу.*

Определение 4. Множеством достижимости по кратным ребрам *для некоторой кратной вершины x назовем множество R_x^k всех вершин y таких, что существует путь из x в y, проходящий только по кратным ребрам.*

Определение 5. Множеством достижимости по обычным ребрам *для некоторой обычной вершины x назовем множество R_x^o всех вершин y таких, что существует путь из x в y, проходящий только по обычным ребрам.*

Очевидно, что $x \in R_x^k$, $x \in R_x^o$. Если $y \in R_x^k$, то $R_y^k = R_x^k$. Если $y \in R_x^o$, то $R_y^o = R_x^o$.

Определение 6. *Целочисленная функция $l(e)$, определенная для всех ребер $e \in E$, является* длиной (весом) ребра *в кратном графе $G(V, E)$, если выполнено следующее:*

1. *$l(e) > 0$ для любого ребра e.*
2. *Если e является кратным или мультиребром, то $l(e_1) = l(e_2) = \ldots = l(e_k)$ и $l(e) = k \cdot l(e_1)$, где $e_1, \ldots, e_k$ – это связанные ребра данного ребра e.*

Тогда *длина кратного пути $S(x, y)$* будет определяться по формуле

$$l(S(x, y)) = \sum_{i=1}^{k} l(S^i(x, y)) = \sum_{i=1}^{k} \sum_{e_j \in S^i(x,y)} l(e_j),$$

при этом может оказаться $e_j = e_p$ $(j \neq p)$, то есть в сумме учитывается каждое повторное вхождение обычного ребра в $S^i(x, y)$.

Задача 1 (кратчайший кратный путь)**.** *В кратном графе $G(V, E)$ требуется найти* кратчайший путь из вершины x в вершину y*, то есть такой путь $S_{\min}(x, y)$, что для любого пути $S(x, y)$ выполнено*

$$l(S_{\min}(x, y)) \leqslant l(S(x, y)).$$

2. Полиномиальный алгоритм поиска кратчайшего кратного пути в произвольном кратном графе

Пусть имеется произвольный взвешенный кратный граф $G(V, E)$ кратности k. Получим полиномиальный алгоритм решения задачи 1 для него. Для этого возьмем алгоритм 2 из статьи [2] и модифицируем его.

Как и ранее, на отдельных шагах алгоритма для нахождения минимальных участков пути, содержащих только обычные ребра, мы будем использовать известный алгоритм Дейкстры (см. [3]). Кроме того, для поиска подмножества минимальных участков пути на шаге 4.4 алгоритма мы будем применять венгерский алгоритм (см. [4]).

Алгоритм ищет кратчайший кратный путь $S_{\min}(x, y)$ между двумя выбранными вершинами x и y. Через e_a^m обозначим мультиребро, инцидентное кратной вершине a. В алгоритме мы будем использовать следующие структуры данных:

- множества достижимости R_a^k и R_b^o;
- множества индексов I_a, ассоциированные с каждым мультиребром e_a^m;
- $G^{ord}(V^{ord}, E^{ord})$ – обычный граф, минимальному пути $S_{\min}^{ord}(x, y)$ в котором будет соответствовать кратчайший кратный путь $S_{\min}(x, y)$ той же длины в исходном графе;
- $S_{mo}(a, b)$ – кратный путь между двумя кратными вершинами, состоящий из мультиребер e_a^m и e_b^m, а также из минимальных обычных путей между соответствующими парами обычных вершин – концов этих мультиребер;
- рейтинговая матрица R^j, используемая при поиске $S_{mo}(a, b)$.

Многие шаги алгоритма будут полностью совпадать с аналогичными шагами алгоритма 2 из статьи [2], поэтому такие шаги мы будем формулировать тезисно за исключением случаев, когда полное описание необходимо для понимания дальнейших действий.

Алгоритм (кратчайший путь в произвольном кратном графе).

1. Найдем все множества достижимости по кратным и обычным ребрам R_a^k и R_b^o. Пронумеруем все множества R_b^o в произвольном порядке от 1 до n и обозначим их через $R_1, \ldots, R_n$.

2. Проверим выполнение критерия существования кратного пути между вершинами x и y. Если критерий не выполнен, выходим из алгоритма.

3. Для каждого мультиребра $e_a^m = \{a, \{a_1, \ldots, a_k\}\}$ сформируем мультимножество индексов $I_a = \{i_1, \ldots, i_k\}$ таким образом, что каждый i_p равен номеру множества достижимости, в которое попадает a_p: $a_p \in R_{i_p}$. Пусть вершины $a_1, \ldots, a_k$ мультиребра e_a^m пронумерованы в порядке возрастания значений i_p (иначе их можно быстро перенумеровать).

4. Будем строить обычный граф $G^{ord}(V^{ord}, E^{ord})$ следующим образом.

4.1. Для каждой кратной вершины $v \in V$ создадим обычную вершину $v \in V^{ord}$.

4.2. Для каждого кратного ребра $\{u, v\} \in E$ создадим обычное ребро $\{u, v\} \in E^{ord}$ той же длины.

4.3. Для каждой кратной вершины a, инцидентной мультиребру e_a^m, создадим дополнительную обычную вершину $a' \in V^{ord}$ и обычное ребро $\{a, a'\} \in E^{ord}$ длины 1.

4.4. Рассмотрим все пары мультиребер из E^m. Для каждой такой пары $\{e_a^m, e_b^m\}$ $(e_a^m = \{a, \{a_1, \ldots, a_k\}\}$, $e_b^m = \{b, \{b_1, \ldots, b_k\}\})$ проверим равенство $I_a = I_b$. Если оно выполнено, будем поочередно рассматривать все подмножества I_a^j совпадающих индексов из I_a.

Если подмножество I_a^j содержит только один элемент i_p ($I_a^j = \{i_p\}$), с помощью алгоритма Дейкстры найдем кратчайший обычный путь $S_{\min}(a_p, b_p)$, проходящий только по вершинам из R_{i_p}. Обозначим $c_p = b_p$.

Если подмножество I_a^j содержит q одинаковых элементов ($I_a^j = \{i_p, i_{p+1}, \dots, i_{p+q-1}\}$, $i_s = i_t$ для всех $s \in \overline{p, p+q-1}$, $t \in \overline{p, p+q-1}$), найдем с помощью алгоритма Дейкстры q^2 кратчайших обычных путей $S_{\min}(a_s, b_t)$ ($s \in \overline{p, p+q-1}$, $t \in \overline{p, p+q-1}$), проходящих только по вершинам из R_{i_p}. Среди них нужно выбрать q путей, попарно непересекающихся в начальных и конечных вершинах и имеющих при этом минимальную суммарную длину.

Это можно сделать следующим образом. Сопоставим каждому a_s работника с номером s, а каждому b_t – работу с номером t ($s \in \overline{p, p+q-1}$, $t \in \overline{p, p+q-1}$). Далее сформируем *рейтинговую матрицу* R^j размерности $q \times q$, где

$$r_{st} = l\left(S_{\min}(a_s, b_t)\right).$$

Задача выбора q путей требуемого вида свелась к *задаче о назначениях* с рейтинговой матрицей R^j, а эта задача может быть решена при помощи венгерского алгоритма (см. [4]). Классический венгерский алгоритм ищет решение максимального веса, однако в статье [5] показано, что тот же алгоритм можно использовать и при поиске решения минимального веса. В результате в каждой строке и каждом столбце указанной матрицы будет выбран ровно один элемент, а сумма этих элементов будет минимально возможной. Выбранные элементы соответствуют искомым q путям.

Упорядочим выбранные пути по возрастанию номера начальной вершины, конечные вершины обозначим через c_r ($r \in \overline{p, p+q-1}$).

Просмотрев все подмножества, сформируем и запомним кратный путь

$$S_{mo}(a, b) = e_a^m \cup S_{\min}(a_1, c_1) \cup \dots \cup S_{\min}(a_k, c_k) \cup e_b^m,$$

который будет кратчайшим кратным путем без кратных ребер между вершинами a и b. Добавим в E^{ord} обычное ребро $\{a', b'\}$ длины $l(\{a', b'\}) = l\left(S_{mo}(a, b)\right) - 2$.

4.5. Если x – обычная вершина, скопируем ее в V^{ord}. Для всех мультиребер e_a^m, для которых это возможно, находим кратчайший кратный путь $S_{mo}(x, a)$ без кратных ребер как в алгоритме 2 из статьи [2]. Добавим в E^{ord} обычное ребро $\{x, a'\}$ длины $l(\{x, a'\}) = l\left(S_{mo}(x, a)\right) - 1$.

4.6. Если y – обычная вершина, скопируем ее в V^{ord}. Для всех мультиребер e_a^m, для которых это возможно, находим кратчайший кратный путь $S_{mo}(a, y)$ без кратных ребер как в алгоритме 2 из статьи [2]. Добавим в E^{ord} обычное ребро $\{a', y\}$ длины $l(\{a', y\}) = l\left(S_{mo}(a, y)\right) - 1$.

4.7. Если обе вершины x и y обычные и $y \in R_x^o$, находим кратчайший путь $S'_{\min}(x, y)$, проходящий только по вершинам из R_x^o. Добавим в E^{ord} обычное ребро $\{x, y\}$ длины $l(S'_{\min}(x, y))$.

5. Найдем кратчайший путь $S_{\min}^{ord}(x, y)$ в графе $G^{ord}(V^{ord}, E^{ord})$ с помощью алгоритма Дейкстры.

6. Преобразуем путь $S_{\min}^{ord}(x, y)$ в графе $G^{ord}(V^{ord}, E^{ord})$ в искомый кратчайший кратный путь $S_{\min}(x, y)$ в графе $G(V, E)$ как в алгоритме 2 из статьи [2].

Применимость полученного алгоритма для задачи 1 в случае произвольного кратного графа следует из применимости алгоритма 2 из статьи [2], которая была обоснована в указанной статье. Действительно, в новом алгоритме меняется лишь принцип выбора путей при построении графа $G^{ord}(V^{ord}, E^{ord})$, условия выбора остаются прежними.

Алгоритм 2 из статьи [2] был экспоненциальным из-за того, что на шаге 4.4 могла возникнуть ситуация перебора $q!$ (а в худшем случае – $k!$) вариантов подмножеств кратчайших путей. Однако в новом алгоритме мы для этой цели производим полиномиальное сведение подзадачи выбора к задаче о назначениях и решаем задачу о назначениях с помощью венгерского алгоритма. В статье [5] показана полиномиальность венгерского алгоритма, откуда и следует полиномиальность нашего алгоритма равно как и полиномиальность задачи 1 для произвольного кратного графа.

Отметим, что алгоритм универсален и применим для любого кратного графа, в том числе для делимого графа. Однако для делимого графа алгоритм 1 из статьи [2] будет более эффективным, поскольку он учитывает специфику построения такого графа.

3. Пример работы алгоритма

Продемонстрируем работу алгоритма. Для этого рассмотрим граф кратности 3, показанный на рис. 1. Для лучшего восприятия на рисунке подписаны только номера вершин без буквы “x”. На ребрах шрифтом Courier New отмечены длины. Кратные ребра показаны жирными линиями, а мультиребра – расщепляющимися на 3 части линиями. Обычные вершины показаны серым.

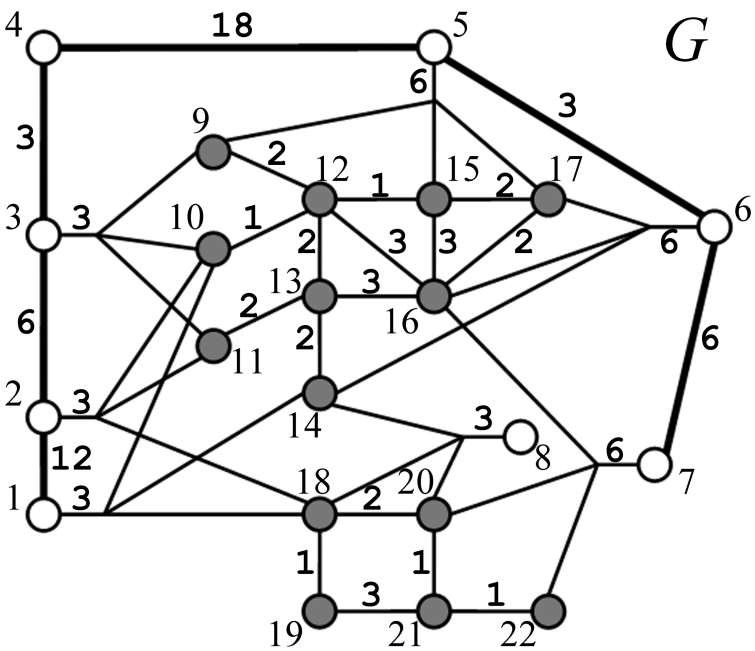

Fig. 1. Weighted graph of multiplicity 3

Рис. 1. Взвешенный граф кратности 3

Будем искать кратчайший кратный путь $S_{\min}(x_1, x_8)$. Вершины x_1 и x_8 кратные и граф $G(V, E)$ связен, поэтому такой путь существует (следует из критерия существования кратного пути – теорема 3 из статьи [1]). Определим и пронумеруем множества достижимости по обычным ребрам:

$$R_1 = R^o_{x_9} = \{x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}\}, \quad R_2 = R^o_{x_{18}} = \{x_{18}, x_{19}, x_{20}, x_{21}, x_{22}\}.$$

Множества индексов I_a для мультиребер тогда определяются так:

$$I_{x_1} = \{1, 1, 2\}, \quad I_{x_2} = \{1, 1, 2\}, \quad I_{x_3} = \{1, 1, 1\}, \quad I_{x_5} = \{1, 1, 1\}, \quad I_{x_6} = \{1, 1, 1\}, \quad I_{x_7} = \{1, 2, 2\}, \quad I_{x_8} = \{1, 2, 2\}.$$

Будем теперь строить обычный граф $G^{ord}(V^{ord}, E^{ord})$ (шаг 4 алгоритма). Шаги 4.1–4.3 реализуются тривиальным образом, шаги 4.5–4.7 не потребуются, так как путь строится между двумя кратными вершинами. Наибольшую сложность представляет шаг 4.4, рассмотрим его подробно. Здесь основные вычисления связаны с поиском кратных путей $S_{mo}(a, b)$ для подходящих пар мультиребер. Таких пар будет пять.

Сначала рассмотрим $e^m_{x_1} = \{x_1, \{x_{10}, x_{14}, x_{18}\}\}$ и $e^m_{x_2} = \{x_2, \{x_{10}, x_{11}, x_{18}\}\}$. Для подмножества $I^1_{x_1} = \{1, 1\}$ найдем кратчайшие пути между всеми парами обычных вершин, соответствующих этому

подмножеству, и получим рейтинговую матрицу (выделенные элементы являются решением задачи о назначениях):

$$R^1 = \begin{array}{c|cc} & x_{10} & x_{11} \\ \hline x_{10} & \boxed{0} & 5 \\ x_{14} & 5 & \boxed{4} \end{array}$$

Таким образом, в $S_{mo}(x_1, x_2)$ нужно включить $S_{\min}(x_{10}, x_{10})$ и $S_{\min}(x_{14}, x_{11})$. Для подмножества $I^2_{x_1} = \{2\}$ достаточно взять пустой кратчайший путь $S_{\min}(x_{18}, x_{18})$ и добавить его в $S_{mo}(x_1, x_2)$. Получаем:

$$S_{mo}(x_1, x_2) = e^m_{x_1} \cup \varnothing \cup \big(\{x_{14}, x_{13}\},\ \{x_{13}, x_{11}\}\big) \cup \varnothing \cup e^m_{x_2}, \quad l\,(S_{mo}(x_1, x_2)) = 10.$$

Далее рассмотрим $e^m_{x_3} = \{x_3, \{x_9, x_{10}, x_{11}\}\}$ и $e^m_{x_5} = \{x_5, \{x_9, x_{15}, x_{17}\}\}$. Для единственного подмножества $I^1_{x_3} = \{1, 1, 1\}$ найдем кратчайшие пути между всеми парами обычных вершин, соответствующих этому подмножеству, и получим рейтинговую матрицу:

$$R^1 = \begin{array}{c|ccc} & x_9 & x_{15} & x_{17} \\ \hline x_9 & \boxed{0} & 3 & 5 \\ x_{10} & 3 & \boxed{2} & 4 \\ x_{11} & 6 & 5 & \boxed{7} \end{array}$$

В $S_{mo}(x_3, x_5)$ нужно включить $S_{\min}(x_9, x_9)$, $S_{\min}(x_{10}, x_{15})$ и $S_{\min}(x_{11}, x_{17})$. Получаем:

$$S_{mo}(x_3, x_5) = e^m_{x_3} \cup \varnothing \cup \big(\{x_{10}, x_{12}\},\ \{x_{12}, x_{15}\}\big) \cup \big(\{x_{11}, x_{13}\},\ \{x_{13}, x_{16}\},\ \{x_{16}, x_{17}\}\big) \cup e^m_{x_5}, \quad l\,(S_{mo}(x_3, x_5)) = 18.$$

Для пары $e^m_{x_3} = \{x_3, \{x_9, x_{10}, x_{11}\}\}$ и $e^m_{x_6} = \{x_6, \{x_{14}, x_{16}, x_{17}\}\}$ то же подмножество $I^1_{x_3} = \{1, 1, 1\}$ приводит к рейтинговой матрице:

$$R^1 = \begin{array}{c|ccc} & x_{14} & x_{16} & x_{17} \\ \hline x_9 & 6 & \boxed{5} & 5 \\ x_{10} & 5 & 4 & \boxed{4} \\ x_{11} & \boxed{4} & 5 & 7 \end{array}$$

В $S_{mo}(x_3, x_6)$ нужно включить $S_{\min}(x_9, x_{16})$, $S_{\min}(x_{10}, x_{17})$ и $S_{\min}(x_{11}, x_{14})$. Получаем:

$$S_{mo}(x_3, x_6) = e^m_{x_3} \cup \big(\{x_9, x_{12}\},\ \{x_{12}, x_{16}\}\big) \cup \big(\{x_{10}, x_{12}\},\ \{x_{12}, x_{15}\},\ \{x_{15}, x_{17}\}\big) \cup \big(\{x_{11}, x_{13}\},\ \{x_{13}, x_{14}\}\big) \cup e^m_{x_6},$$

$$l\,(S_{mo}(x_3, x_6)) = 22.$$

Стоит отметить, что $S_{mo}(x_3, x_6)$ может быть получен и в альтернативном виде — с использованием $S_{\min}(x_9, x_{17})$, $S_{\min}(x_{10}, x_{16})$ и $S_{\min}(x_{11}, x_{14})$.

Еще одна пара $e^m_{x_5} = \{x_5, \{x_9, x_{15}, x_{17}\}\}$ и $e^m_{x_6} = \{x_6, \{x_{14}, x_{16}, x_{17}\}\}$ с тем же множеством индексов дает рейтинговую матрицу:

$$R^1 = \begin{array}{c|ccc} & x_{14} & x_{16} & x_{17} \\ \hline x_9 & \boxed{6} & 5 & 5 \\ x_{15} & 5 & \boxed{3} & 2 \\ x_{17} & 7 & 2 & \boxed{0} \end{array}$$

В $S_{mo}(x_5, x_6)$ нужно включить $S_{\min}(x_9, x_{14})$, $S_{\min}(x_{15}, x_{16})$ и $S_{\min}(x_{17}, x_{17})$. Получаем:

$$S_{mo}(x_5, x_6) = e^m_{x_5} \cup \big(\{x_9, x_{12}\},\ \{x_{12}, x_{13}\},\ \{x_{13}, x_{14}\}\big) \cup \big(\{x_{15}, x_{16}\}\big) \cup \varnothing \cup e^m_{x_6}, \quad l\,(S_{mo}(x_5, x_6)) = 21.$$

Рассмотрим последнюю пару $e^m_{x_7} = \{x_7, \{x_{16}, x_{20}, x_{22}\}\}$ и $e^m_{x_8} = \{x_8, \{x_{14}, x_{18}, x_{20}\}\}$. Для подмножества $I^1_{x_7} = \{1\}$ нужно взять кратчайший путь $S_{\min}(x_{16}, x_{14})$ и добавить его в $S_{mo}(x_7, x_8)$. Для подмножества $I^2_{x_7} = \{2, 2\}$, как и выше, выполним сведение к задаче о назначениях:

$$R^2 = \begin{array}{c|cc} & x_{18} & x_{20} \\ \hline x_{20} & 2 & \boxed{0} \\ x_{22} & \boxed{4} & 2 \end{array}$$

Таким образом, в $S_{mo}(x_7, x_8)$ нужно включить $S_{\min}(x_{20}, x_{20})$ и $S_{\min}(x_{22}, x_{18})$ (но есть и альтернативный вариант с двумя путями длины 2). Получаем:

$$S_{mo}(x_7, x_8) = e^m_{x_7} \cup \big(\{x_{16}, x_{13}\},\ \{x_{13}, x_{14}\}\big) \cup \varnothing \cup \big(\{x_{22}, x_{21}\},\ \{x_{21}, x_{20}\},\ \{x_{20}, x_{18}\}\big) \cup e^m_{x_8}, \quad l\,(S_{mo}(x_7, x_8)) = 18.$$

В итоге выполнения шага 4 алгоритма будет получен обычный граф $G^{ord}(V^{ord}, E^{ord})$, показанный на рис. 2. Вершины кратчайшего пути $S^{ord}_{\min}(x_1, x_8)$ (шаг 5) показаны серым. Длина этого пути $l\left(S^{ord}_{\min}(x_1, x_8)\right) = 61$.

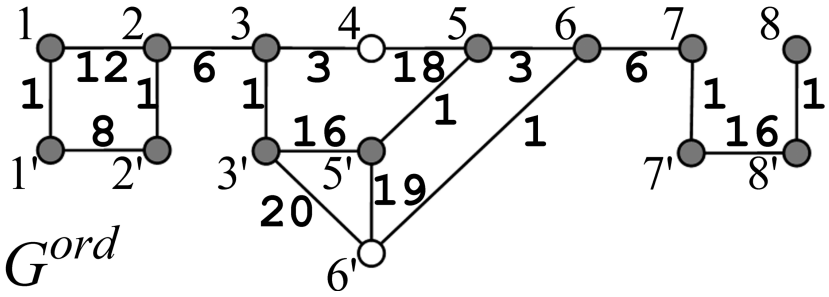

Fig. 2. Ordinary graph G^{ord} and the shortest path

Рис. 2. Обычный граф G^{ord} и кратчайший путь

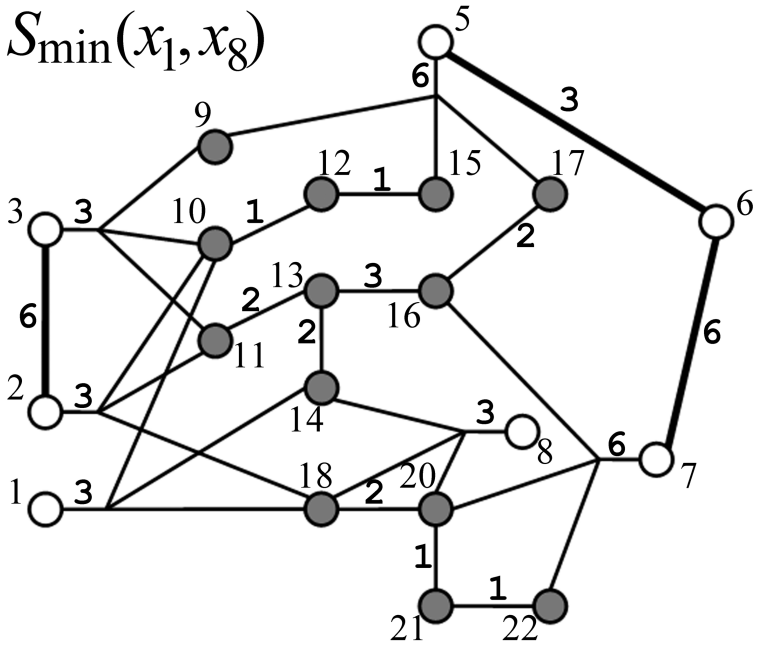

Fig. 3. The shortest path in the multiple graph G

Рис. 3. Кратчайший путь в кратном графе G

Применяя шаг 6, преобразуем путь $S^{ord}_{\min}(x_1, x_8)$ в искомый кратчайший кратный путь $S_{\min}(x_1, x_8)$ (рис. 3):

$$S_{\min}(x_1, x_8) = \left(S_{mo}(x_1, x_2),\ \{x_2, x_3\},\ S_{mo}(x_3, x_5),\ \{x_5, x_6\},\ \{x_6, x_7\},\ S_{mo}(x_7, x_8)\right),$$

$$l\left(S_{\min}(x_1, x_8)\right) = l\left(S^{ord}_{\min}(x_1, x_8)\right) = 61.$$

Заметим, что каждое из обычных ребер $\{x_{11}, x_{13}\}$, $\{x_{13}, x_{14}\}$ и $\{x_{13}, x_{16}\}$ проходится дважды на разных участках полученного кратного пути, причем в противоположных направлениях, однако для кратных путей повторение обычных ребер допустимо.

References

[1] A. V. Smirnov, "The Shortest Path Problem for a Multiple Graph", *Automatic Control and Computer Sciences*, vol. 52, no. 7, pp. 625–633, 2018. DOI: 10.3103/S0146411618070234.

[2] A. V. Smirnov, "The Polynomial Algorithm of Finding the Shortest Path in a Divisible Multiple Graph", *Modeling and Analysis of Information Systems*, vol. 29, no. 4, pp. 372–387, 2022. DOI: 10.18255/1818-1015-2022-4-372-387.

[3] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959. DOI: 10.1007/BF01386390.

[4] H. W. Kuhn, "The Hungarian method for the assignment problem", *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. DOI: 10.1002/nav.3800020109.

[5] J. Munkres, "Algorithms for the Assignment and Transportation Problems", *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957. DOI: 10.1137/0105003.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 1, 2023
journal homepage: www.mais-journal.ru

COMPUTER SYSTEM ORGANIZATION

Construction of an Adaptive Motion Control System Optimal Information Exchange Scheme for a Group of Unmanned Aerial Vehicles

L. N. Kazakov[1], E. P. Kubyshkin[1], D. E. Paley[1]

DOI: 10.18255/1818-1015-2023-1-16-26

[1]P. G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 49J15
Research article
Full text in Russian

Received December 3, 2022
After revision February 6, 2023
Accepted February 8, 2023

The paper considers the problem of modeling an adaptive control system information exchange for a group of unmanned aerial vehicles (UAVs). The movement of the UAV group is carried out in accordance with the adaptive algorithm for optimal control. Optimal controls are constructed to provide a minimum of the total energy expended. The parameters of the mathematical model of the movement of the UAV group are refined during the flight in accordance with changing external conditions. In accordance with this, the control actions are specified. This task requires significant computing resources and imposes special requirements on the information exchange system between the UAV and the control point. A scheme of information exchange between the UAV and the control point is proposed, which makes it possible to calculate the optimal parameters of the transmitting devices.

Keywords: information exchange; optimal control; optimal trajectory; group of unmanned aerial vehicles

INFORMATION ABOUT THE AUTHORS

Leonid Nikolaevich Kazakov correspondence author	orcid.org/0000-0002-9348-4440. E-mail: kazakov@uniyar.ac.ru Head of the Department of Radio Engineering Systems, Doctor of Science, Professor.
Evgeniy Pavlovich Kubyshkin	orcid.org/0000-0003-1796-0190. E-mail: kubysh.e@yandex.ru Professor of the Department of Mathematical Modeling, Doctor of Science, Professor.
Dmitry Ezrovich Paley	orcid.org/0000-0002-9184-3286. E-mail: dpaley@list.ru Associate Professor, PhD.

Funding: This work was carried out within the framework of a development programme for the Regional Scientific and Educational Mathematical Center of the Yaroslavl State University with financial support from the Ministry of Science and Higher Education of the Russian Federation (Agreement on provision of subsidy from the federal budget No. 075-02-2022-886) and funded by YSU Programme according to the research project №P2-K-1-G-4/2021 P2-GM-2021.

For citation: L. N. Kazakov, E. P. Kubyshkin, and D. E. Paley, “Construction of an Adaptive Motion Control System Optimal Information Exchange Scheme for a Group of Unmanned Aerial Vehicles”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 16-26, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 1, 2023
сайт журнала: www.mais-journal.ru

COMPUTER SYSTEM ORGANIZATION

Построение оптимальной схемы информационного обмена системы адаптивного управления движением группы беспилотных летательных аппаратов

Л. Н. Казаков[1], Е. П. Кубышкин [1], Д. Э. Палей[1]

DOI: 10.18255/1818-1015-2023-1-16-26

[1]Ярославский государственный университет им. П. Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 517.977.51
Научная статья
Полный текст на русском языке

Получена 3 декабря 2022 г.
После доработки 6 февраля 2023 г.
Принята к публикации 8 февраля 2023 г.

В работе рассмотрена задача моделирования информационного обмена адаптивной системы управления движением группы беспилотных летательных аппаратов (БЛА). Движение группы БЛА осуществляется в соответствии с адаптивным алгоритмом оптимального управления пространственной перестройкой. Оптимальные управления строятся обеспечивающими минимум общей затрачиваемой энергии. Параметры математической модели движения группы БЛА уточняются в процессе полета в соответствии с изменяющимися внешними условиями. В соответствии с этим уточняются управляющие воздействия. Это требует значительных вычислительных ресурсов и накладывает особые требования на систему информационного обмена между БЛА и пунктом управления. Предложена схема информационного обмена между БЛА и пунктом управления, позволяющая рассчитать оптимальные параметры передающих устройств.

Ключевые слова: информационный обмен; оптимальное управление; оптимальная траектория; группа беспилотных летательных аппаратов

ИНФОРМАЦИЯ ОБ АВТОРАХ

Леонид Николаевич Казаков автор для корреспонденции	orcid.org/0000-0002-9348-4440. E-mail: kazakov@uniyar.ac.ru Заведующий кафедрой радиотехнических систем, д-р тех. наук, профессор.
Евгений Павлович Кубышкин	orcid.org/0000-0003-1796-0190. E-mail: kubysh.e@yandex.ru Профессор кафедры математического моделирования, д-р физ.-мат. наук, профессор.
Дмитрий Эзрович Палей	orcid.org/0000-0002-9184-3286. E-mail: dpaley@list.ru Доцент кафедры радиотехнических систем, канд. тех. наук.

Финансирование: Работа выполнена в рамках реализации программы развития регионального научно-образовательного математического центра (ЯрГУ) при финансовой поддержке Министерства науки и высшего образования РФ (соглашение о предоставлении из федерального бюджета субсидии № 075-02-2022-886) и Программы развития ЯрГУ, проект №П2-К-1-Г-4/2021 П2-ГМ4-2021.

Для цитирования: L. N. Kazakov, E. P. Kubyshkin, and D. E. Paley, “Construction of an Adaptive Motion Control System Optimal Information Exchange Scheme for a Group of Unmanned Aerial Vehicles”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 16-26, 2023.

Введение

Управление движением группы беспилотных летательных аппаратов (БЛА) представляет собой весьма непростую задачу. Она содержит как минимум две основные составляющие – построение алгоритма управления и построение системы информационного обмена между БЛА и пунктом управления, включающей в себя как алгоритмы информационного обмена, так и аппаратные средства. Решению первой части посвящено большое количество работ. Отметим основные подходы, используемые при решении первой части задачи управления. В работах [1—4] рассматриваются методы на основе построения графов. Подобные методы обычно используют в тех случаях, когда информация об окружающей среде является статической и известна полностью. Другим эффективным подходом планирования траекторий являются методы на основе «клеточной декомпозиции» [5—8]. Основная идея таких методов – планирование пути в среде с препятствиями на основе дискретизации окружающей среды. Сетка проста в использовании, и её легко создать и поддерживать. Однако метод сеток имеет серьезный недостаток: увеличение трудоемкости при уменьшении шагов сетки.

В [9—11] управления строятся на основе метода потенциальных функций. Алгоритм оптимального управления основывается на движении вдоль линий векторного поля, потенциальная функция которого отражает конфигурацию препятствий и их форму, а также цель движения. Отметим, что среди методов потенциальных полей самым известным является метод искусственных потенциалов (artificial potential field, APF) [11]. Его алгоритм прост, имеет низкую вычислительную сложность и высокую эффективность реализации. Векторное поле разделяется на две составляющие: цель движения представляется притягивающим векторным полем, в то время как препятствия — отталкивающим векторным полем. В [12] рассматривается задача стабилизации БЛА.

Решение задачи оптимального выбора траектории существенно зависит от рассматриваемой модели и ее параметров. При этом следует учитывать, что параметры модели в общем случае являются функциями времени. Т.е. всю систему необходимо рассматривать как адаптивную. В частности, перед синтезом оптимального управления необходимо решить задачу идентификации параметров системы. В настоящее время существует большое количество различных подходов к формированию адаптивного управления. Широко применяются классические методы с «эталонной моделью» [13, 14]. Можно также отметить L1 адаптивное управление [15, 16], которое позволяет отделить оценку модели от управления. Задача планирования пути в сложной окружающей среде может так же решаться, как оптимизационная задача. Для этого движение объекта надо представить как движение динамической системы в рамках некоторой модели. Препятствия будут описываться некоторыми ограничениями, а качество допустимой траектории соответственно должно оцениваться некоторым функционалом. В результате возникает задача оптимального управления. Решение такой задачи обеспечивает траекторию объекта в обход препятствий и позволяет выбрать лучший вариант с точки зрения функционала качества. Существует несколько подходов к решению таких задач. Наиболее популярными из них являются методы на основе нелинейного программирования [17, 18].

В настоящей работе рассмотрена задача моделирования информационного обмена адаптивной системы управления движением группы беспилотных летательных аппаратов (БЛА), предложенной в [19, 20]. Движение БЛА осуществляется в соответствии с адаптивным алгоритмом оптимального управления пространственной перестройкой. Параметры математической модели движения группы БЛА уточняются в процессе полета в соответствии с изменяющимися внешними условиями и соответственно происходит уточнение управляющих воздействий. Использование адаптивного алгоритма требует значительных вычислительных ресурсов. Это накладывает особые требования на систему информационного обмена между БЛА и пунктом управления. Оптимальные управления строятся доставляя минимум общей затрачиваемой энергии. Предложена схема

информационного обмена между БЛА и пунктом управления, позволяющая рассчитать оптимальные параметры передающих устройств.

1. Алгоритм адаптивного управления движением БЛА

Рассмотрим сначала кратко алгоритм адаптивного управления движением группы беспилотных летательных аппаратов (БЛА), на основе которого строится система информационного обмена (трафика) между БЛА и пунктами управления.

Рассматривается группа из n БЛА, положение которых относительно некоторой инерциальной системы координат $OXYZ$ определяется системой векторов

$$q_j(t) = (x_j(t), y_j(t), z_j(t)),\ j = 1, 2, \ldots, n.$$

Предполагается, что группа БЛА, находясь в момент времени t_0 в точке пространства с координатами q_{j0} и имея скорости $\dot{q}_{j0}$, должна за время $T > 0$ перестроится в точки с координатами q_{jT} и иметь соответственно скорости $\dot{q}_{jT}$. При этом расстояние между отдельными БЛА в группе не должно быть менее $d_{jk} > 0$ и более D_{jk} $j, k = 1, \ldots, n$. Уравнения движения БЛА в группе и динамическая модель полетной перестройки БЛА будут иметь следующий вид [21]

$$m_j \ddot{q}_j + f_j(q_j, \dot{q}_j) = U_j(t),\quad t_0 < t < t_1 = t_0 + T, \tag{1}$$

$$q_j(t_0) = q_{j0},\ \dot{q}_j(t_0) = \dot{q}_{j0},\ q_j(t_1) = q_{j1},\ \dot{q}_j(t_1) = \dot{q}_{j1}, \tag{2}$$

$$d_{jk} \le |q_j(t) - q_k(t)| \le D_{jk},\quad j, k = 1, \ldots, n. \tag{3}$$

В (1) m_j – масса j -го БЛА, функция $f_j(q_j, \dot{q}_j)$ характеризует совокупность внешних сил, действующих на j -й БЛА, включающих в себя вес БЛА, внешнее трение, ветровую нагрузку и другие внешние силы, $U_j(t) = (U_{j1}(t), U_{j2}(t), U_{j3}(t))$ – вектор силового управления БЛА вдоль осей соответствующих координат, $|^*|$ – евклидово расстояние между точками. На практике выражения и значения функций $f_j(q_j, \dot{q}_j)$ априори не всегда точно известны. Будем предполагать, что они являются дифференцируемыми функциями и что их постоянные составляющие отнесены к функциям $U_j(t)$, поэтому можно считать $f_j(0, 0) = 0$. Предполагается, что между собой БЛА в группе не взаимодействуют.

Энергия силового управления j-го БЛА определяется интегралом

$$J(U_j(t)) = \int_0^T (U_j(t), U_j(t))dt, \tag{4}$$

где $(.,.)$ – скалярное произведение в R^3, соответственно

$$J(U(t)) = \sum_{j=1}^{n} J(U_j(t)) \tag{5}$$

общая энергия силового управления группой БЛА. Предполагается, что $U_{jp}(t) \in L_2(0, T)$ – пространство интегрируемых с квадратом функций. Отметим, что выражение (5) может быть более сложной положительно определенной квадратичной формой. Принципиального значения это не имеет.

Рассматриваются следующие задачи оптимального управления при построении траекторий движения группы БЛА.

Задача 1. Найти управления $U_j(t), j = 1, \ldots, n$, обеспечивающие перевод траекторий системы уравнений (1) за время $T > 0$ из начального положения (2) в конечное с учетом условий (3) и доставляющие минимум функционалу (5).

Задача 2. Найти управления $U_j(t), j = 1, \ldots, n$, при условии $J(U_j) \le L_j, L_j > 0$, обеспечивающее перевод траекторий системы уравнений (1) из начального положения (2) в конечное с учетом условий (3) за минимальное время $T > 0$.

В рамках определенных функций $f_j(q_j, \dot{q}_j)$ в работах [19, 20] предложен алгоритм решения указанных задач. Алгоритм итерационный и требует значительных вычислительных ресурсов.

Для вычисления оптимального управления $U_j(t)$ необходимо выбрать конкретный вид функций $f_j(q_j, \dot{q}_j)$. Это делается как на этапе моделирования, так и на этапе оценки внешних воздействий перед синтезом $U_j(t)$. Очевидно, что точно задать $f_j(q_j, \dot{q}_j)$ не всегда представляется возможным и модель (1) может стать ошибочна прямо во время полета.

Например, сила ветра в начале маршрута может существенно отличаться от силы ветра на различных его этапах, или за счет дождя и неравномерного расхода топлива может измениться масса БЛА и т. д. В силу этого $U_j(t)$, вычисленное по неверной модели, перестает удовлетворять решению задачи. Рассмотрим задачу уточнения функций $f_j(q_j, \dot{q}_j)$ прямо во время полета БЛА. Такой алгоритм построения $U_j(t)$ назовем адаптивным. В соответствии с этим будет строиться система информационного обмена.

Управления $U_j(t)$ формируются динамически исходя из заданных критериев и параметров модели (1). Предположим, что $U_j(t)$ синтезировано в некоторый момент времени $t = t_0$. Далее управление поступает на вход реальной системы. Если модель (1) верна, то реальная траектория $p_j(t)$, при $t > t_0$, будет совпадать с расчетной $q_j(t)$ при условии $p_j(t_0) = q_j(t_0)$. Если модель (1) неверна и функции $f_j(q_j, \dot{q}_j)$ не соответствуют действительности, то между реальной и расчетной траекториями возникнет ошибка

$$p_j(t) - q_j(t) = \Delta_j(t). \tag{6}$$

Если ошибка (6) не равна нулю, то функции $f_j(q_j, \dot{q}_j)$ нуждаются в корректировке. Решим задачу уточнения функций $f_j(q_j, \dot{q}_j)$ на основе динамической ошибки $\Delta_j(t)$. Будем считать, что фактическая траектория движения $p_j(t)$, а также ее производные доступны для измерения. Для уточнения параметров модели $f_j(q_j, \dot{q}_j)$ воспользуемся методами градиентного спуска.

Корректировку функций $f_j(q_j, \dot{q}_j)$ будем проводить для каждого БЛА в отдельности. Рассмотрим сначала линейную модель и запишем уравнение (1) в следующей форме:

$$a_0\ddot{q}_j + a_1\dot{q}_j + a_2 q_j = U_j(t), \tag{7}$$

где для удобства изложения обозначили $m_j = a_0$, a_1, a_2 – коэффициенты при линейных составляющих функции $f_j(q_j, \dot{q}_j)$. Для уточнения коэффициентов (7) воспользуемся известным подходом на основе «идентификации методом обучения» [13, 14].

Пусть на основе коэффициентов a_k рассчитано оптимальное управление $U_j(t)$. Далее $U_j(t)$ подадим на вход реальной системы

$$b_0\ddot{p}_j + b_1\dot{p}_j + b_2 p_j = U_j(t), \tag{8}$$

где b_k – неизвестные коэффициенты. Решение $p_j(t)$ для реальной системы с оптимальным управлением $U_j(t)$ получено в результате измерений. Подставим фактическое решение $p_j(t)$ в исходную модель (7)

$$a_0\ddot{p}_j + a_1\dot{p}_j + a_2 p_j = A_j(t). \tag{9}$$

Если коэффициенты $b_k \neq a_k$, то $q_j(t) \neq p_j(t)$ и для истинности (9) управление должно измениться. Будем считать, что решение $p_j(t)$ является решением уравнения (9) , если управление (правая часть (9)) равно $A_j(t)$. Вычислим разницу в управлениях

$$e_j(t) = A_j(t) - U_j(t) = (a_0 - b_0)\ddot{p}_j + (a_1 - b_1)\dot{p}_j + (a_2 - b_2)p_j. \tag{10}$$

Задача будет решена, если ошибка $e_j(t)$ с течением времени будет стремиться к нулю. Очевидно, что в этом случае также будет стремиться к нулю ошибка $\Delta_j(t)$. Будем рассматривать коэффициенты $a_k =$

$a_k(t)$ как функции времени t. Необходимо изменить $a_k(t)$ таким образом, чтобы минимизировать $e_j(t)$. Очевидно, что ошибка равна нулю при выполнении $a_k(t) = b_k$. Введем функцию качества

$$V(t) = ((a_0(t) - b_0)^2 + (a_1(t) - b_1)^2 + (a_2(t) - b_2)^2)/2. \tag{11}$$

Заметим, что $V(t)$ положительно определенная, $V(t) = 0$, при $a_k(t) = b_k$, т. е. минимум $V(t)$ соответствует минимуму $e_j(t)$. Вычислим

$$\frac{dV}{dt} = (a_0(t) - b_0)\frac{da_0}{dt} + (a_1(t) - b_1)\frac{da_1}{dt} + (a_2(t) - b_2)\frac{da_2}{dt}. \tag{12}$$

Найдем условия, при которых производная (12) по времени будет меньше нуля. Предварительно отметим, что производная ошибки $e_j(t)$ по a_k из (10) равна

$$\frac{de_j(t)}{da_k} = \frac{d^k p_j}{dt^k}. \tag{13}$$

Учитывая требование отрицательности производной, выберем

$$\frac{da_k}{dt} = -e_j(t)\frac{d^k p_j}{dt^k}, \quad a_k(t_0) = a_k. \tag{14}$$

При выполнении (14) несложно видеть из (13), что

$$\frac{dV}{dt} = -((a_0(t) - b_0)\frac{d^2 p_j}{dt^2} + (a_1(t) - b_1)\frac{dp_j}{dt} + (a_2(t) - b_2)p_j)e_j(t) = -e_j(t)^2. \tag{15}$$

Таким образом, при выполнении равенства (14), положительно определенная функция качества $V(t)$ имеет отрицательную производную, т.е. постоянно убывает с течением времени. Соответственно значения коэффициентов модели $a_k(t)$ стремятся к параметрам реальной системы b_k. Равенства (14) позволяют получить выражение для вычисления $a_k(t)$

$$\frac{da_k}{dt} = -e_j(t)\frac{d^k p_j}{dt^k} = -\sum_{m=0}^{2}(a_m\frac{d^m p_j}{dt^m} - U_j(t))\frac{d^k p_j}{dt^k}, \quad a_k(t_0) = a_k, \quad k = 0, 1, 2. \tag{16}$$

Система (16) – это система из трех неоднородных дифференциальных уравнений с переменными коэффициентами. В аналитическом виде решение (16) получить не всегда возможно, кроме того, $p_j(t)$ – это наблюдение реальной траектории динамической системы. С другой стороны, методы численных решений систем дифференциальных уравнений достаточно развиты и могут быть применены к вычислению $a_k(t)$.

Распространим описанный выше подход на случай, когда коэффициенты уравнения (7) нелинейные функции, считая, что каждый может быть представлен в виде степенного ряда

$$a_k(q_j, \dot{q}_j) = \sum_{s=0}^{\infty}(a_{ks}q_j^s + a^*_{ks}\dot{q}_j^s), \; a_k = a_{k0} + a^*_{k0}, \; k = 0, 1, 2. \tag{17}$$

Соответственно, коэффициенты реальной системы дифференциальных уравнений так же представим в виде ряда

$$b_k(p_j, \dot{p}_j) = \sum_{s=0}^{\infty}(b_{ks}p_j^s + b^*_{ks}\dot{p}_j^s), \; b_k = b_{k0} + b^*_{k0}, \; k = 0, 1, 2. \tag{18}$$

На практике количество коэффициентов в разложениях (17), (18) выбирается конечным в соответствии с требованиями точности и конкретным видом функций $a_k(q_j, \dot{q}_j)$, $b_k(p_j, \dot{p}_j)$.

С учетом (17), (18) запишем выражение для ошибки, аналогичное (10)

$$e_j(t) = A_j(t) - U_j(t) = \sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks} - b_{ks})p_j^s + (a^*_{ks} - b^*_{ks})(\frac{dp_j}{dt})^s)\frac{d^k p_j}{dt^k}, \tag{19}$$

где по-прежнему $p_j(t)$ – фактическая траектория движения системы, $U_j(t)$ – синтезированное управляющее воздействие на входе системы, $A_j(t)$ – управляющее воздействие, которое делает истинной модель для решения $p_j(t)$.

Далее будем считать, что коэффициенты a_{ks}, a^*_{ks} зависят от времени и аналогично (11) введем функцию качества

$$V(t) = \sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks}(t) - b_{ks})^2 + (a^*_{ks}(t) - b^*_{ks})^2)/2. \tag{20}$$

Функция $V(t)$ положительно определенная, $V(t) = 0$ при $a_{ks}(t) = b_{ks}$, $a^*_{ks}(t) = b^*_{ks}$, т.е. минимум $V(t)$ соответствует минимуму $e_j(t)$. Вычислим

$$\frac{dV}{dt} = \sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks}(t) - b_{ks})\frac{da_{ks}}{dt} + (a^*_{ks}(t) - b^*_{ks})\frac{da^*_{ks}}{dt}). \tag{21}$$

Чтобы выполнить требование отрицательности производной (21), положим

$$\frac{da_{ks}}{dt} = -e_j(t)p_j^s\frac{d^k p_j}{dt^k}, \quad \frac{da^*_{ks}}{dt} = -e_j(t)(\frac{dp_j}{dt})^s\frac{d^k p_j}{dt^k}, \quad a_{ks}(t_0) = a_{ks}, \; a^*_{ks}(t_0) = a^*_{ks}. \tag{22}$$

В результате с учетом (19) получим равенство

$$\frac{dV}{dt} = \sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks}(t) - b_{ks})\frac{da_{ks}}{dt} + (a^*_{ks}(t) - b^*_{ks})\frac{da^*_{ks}}{dt}) = \tag{23}$$

$$= -e_j(t)\sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks}(t) - b_{ks})p_j^s + (a^*_{ks}(t) - b_{ks}*)(\frac{dp_j}{dt})^s)\frac{d^k p_j}{dt^k} = -e_j^2(t), \tag{24}$$

т. е. при истинности (22) значения коэффициентов модели $a_{ks}(t)$, $a^*_{ks}(t)$ сходятся к параметрам реальной системы b_{ks}, b^*_{ks}. Равенства (22) позволяют получить выражение для вычисления $a_{ks}(t)$, $a^*_{ks}(t)$:

$$\frac{da_{ks}}{dt} = -(A_j(t) - U_j(t))p_j^s\frac{d^k p_j}{dt^k} = \sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks}p_j^s + a^*_{ks}(\frac{dp_j}{dt})^s)(\frac{dp_j}{dt} - U_j(t))p_j^s\frac{d^k p_j}{dt^k}), \; a_{ks}(t_0) = a_{ks}, \tag{25}$$

$$\frac{da^*_{ks}}{dt} = -(A_j(t) - U_j(t))(\frac{dp_j}{dt})^s\frac{d^k p_j}{dt^k} = \sum_{k=0}^{2}\sum_{s=0}^{\infty}((a_{ks}p_j^s + a^*_{ks}(\frac{dp_j}{dt})^s)(\frac{dp_j}{dt} - U_j(t))(\frac{dp_j}{dt})^s\frac{d^k p_j}{dt^k}), \; a^*_{ks}(t_0) = a^*_{ks}. \tag{26}$$

Система (25), (26) является счетной системой обыкновенных дифференциальных уравнений. На практике требуется лишь конечное число коэффициентов разложений (17), которые могут быть найдены из (25), (26) с использованием численных методов. Принципиально (25), (26) не отличается от уравнений (16) и при s=0 переходит в (16). Таким образом, модель (25), (26) является универсальной.

Отметим следующее обстоятельство. Алгоритм построения оптимальных управлений $U_j(t)$ [19, 20] предложен для случая постоянных коэффициентов a_{ks}, a^*_{ks} а из (25), (26) получаем функции $a_{ks}(t), a^*_{ks}(t)$. Здесь можно поступить двумя способами: алгоритм легко обобщается на случай переменных коэффициентов, либо необходимо усреднить коэффициенты, положив

$$a_{ks} = \int_{t_0}^{t_0+T} a_{ks}(t)dt/T, \quad a^*_{ks} = \int_{t_0}^{t_0+T} a^*_{ks}(t)dt/T,$$

где T – временной шаг, выбор которого зависит от конкретной ситуации. Отметим также, что все изложенное легко обобщается на случай, когда в (18) коэффициенты $a^*_{ks} = a^*_{ks}(q_j)$, т.е. являются функциями q_j.

2. Схема информационного обмена алгоритма адаптивного управления движением группы БЛА

Рассмотренный алгоритм требует значительных вычислительных ресурсов, поэтому система информационного обмена между БЛА и центральным устройством управления (ЦУУ) должна учитывать это обстоятельство. Рассмотрим систему информационного обмена при централизованной стратегии управления, при котором ЦУУ о состоянии всех БЛА группы и окружающей среды. ЦУУ производит анализ текущей ситуации и принимает решения о действиях БЛА в группе. Каждый БЛА группы передает на центральное устройство управления данные о текущей траектории движения, а также о состоянии окружающей среды. Центральное устройство управления производит обработку собранных сведений согласно изложенного выше алгоритма и передает каждому БЛА группы команды управления. ЦУУ обычно располагается на земле. Такая форма организации информационного обмена при использовании группы БЛА на больших расстояниях требует наличия на каждом БЛА достаточно мощных приемно-передающих устройств, обеспечивающих надежные каналы связи. Более рациональной выглядит схема организации информационного обмена с одним ведущим. Вычислительные функции осуществляет ЦУУ, которое располагается на земле, а информационный обмен между БЛА и ЦУУ организуется следующим образом. Один БЛА (диспетчер) должен обладать многоканальным приемно-передающим модемом (ППМ), мощность которого должна быть достаточной для связи с ЦУУ. Информационный обмен между БЛА организуется по принципу каждый с каждым (частично) и каждый с диспетчером. В системе управления движением каждый БЛА передает координаты реальной траектории движения (функции $p_j(t)$), вычисленные в строго определенные моменты времени t_k с шагом Δt. Величина шага определяется требуемой точностью вычислений. Это требует временной синхронизации между БЛА, что обеспечивается каналами связи между ними. Величины $p_{jk} = p_j(t_k)$ определяются с помощью навигационной системы (GPS, ГЛОНАСС, инерциальная система навигации). Величины p_{jk} передаются БЛА-диспетчеру и каждому БЛА. Последнее необходимо для работы системы предупреждения непредвиденных столкновений. Такая система должна присутствовать на каждом БЛА, ее описание и построение представляет собой отдельную задачу. Потоки p_{jk} , принимаемые БЛА-диспетчером будут разделены по времени. Это обусловлено разными взаимными расстояниями между БЛА и БЛА-диспетчером. В связи с этим общий информационный поток, поступающий БЛА-диспетчеру можно считать пуассоновским. При этом ППМ БЛА-диспетчера можно рассматривать как многоканальную систему массового обслуживания с ограниченной очередью. В соответствии с этим рассчитаем количество каналов, величину очереди и пропускную способность ППМ БЛА-диспетчера. Пусть λ_j (бит/с, $j = 1, \ldots, n$) – интенсивность информационного потока, поступающего с j- го БЛА БЛА-диспетчеру. Тогда общий информационный поток имеет интенсивность

$$\lambda = \sum_{j=1}^{n} \lambda_j. \tag{27}$$

Предположим, что ППМ имеет m каналов с интенсивностью обслуживания канала μ (бит/с) и очередью на обслуживание в k мест. Очередь необходима, чтобы любой поступающий сигнал был обязательно обслужен. В этом случае относительная пропускная способность ППМ будет иметь вид [22]

$$Q = p_{\text{обс}} = 1 - p_{\text{отк}},$$

где $p_{\text{обс}}$ – вероятность обслуживания сигнала, $p_{\text{отк}}$ – вероятность отказа в обслуживании сигнала,

$$p_{\text{отк}} = \frac{\rho^{m+k}}{m^k m!} p_0,\ \rho = \frac{\lambda}{\mu},\ p_0 = \left(1 + \frac{\rho}{1!} + \frac{\rho^2}{2!} + \ldots + \frac{\rho^m}{m!} + \frac{\rho^{m+1}}{mm!} + \ldots + \frac{\rho^{m+k}}{m^k m!}\right)^{-1}.$$

Следующие характеристики ППМ позволят оптимально выбрать структуру его каналов и величину очереди:

абсолютная пропускная способность

$$A = \mu Q,$$

среднее число занятых каналов

$$m_{\text{З}} = A/\mu = \rho Q,$$

среднее число простаивающих каналов

$$m_{\text{ПР}} = m - m_{\text{З}},$$

коэффициент занятости (использования) каналов

$$K_{\text{З}} = m_{\text{З}}/m,$$

коэффициент простоя каналов

$$K_{\text{ПР}} = 1 - K_{\text{З}},$$

среднее число заявок, находящихся в очереди,

$$L_{\text{ОЧ}} = \frac{\rho^{m+1}}{mm!}\frac{1-(\rho/m)^k(k+1-k\rho/m)}{(1-\rho/m)^2}p_0,$$

а в случае, если $\rho/m = 1$, эта формула принимает другой вид

$$L_{\text{ОЧ}} = \frac{\rho^{m+1}}{mm!}\frac{k(k+1)}{2}p_0,$$

среднее время ожидания в очереди и среднее время пребывания в ППМ определяется выражениями, называемыми формулами Литтла,

$$T_{\text{ОЧ}} = \frac{L_{\text{ОЧ}}}{\lambda}, \quad T_{\text{ППМ}} = \frac{L_{\text{ОЧ}}}{\lambda} + \frac{Q}{\mu}.$$

Возможна схема, при которой функции ЦУУ будет выполнять БЛА-диспетчер с использованием бортовых вычислительных и телекоммуникационных устройств, т.е. будет реализована схема централизованного управления с одним ведущим. Однако в этом случае на бортовые вычислительные устройства ведущего ложится значительная вычислительная нагрузка. При этом обмен информации между БЛА в группе лучше всего организовать по кольцевой схеме. Это выглядит следующим образом. Примем за полный цикл информационного обмена внутри группы БЛА $T_{\text{Ц}}$ с. Такой интервал хорошо синхронизируется с временными метками GPS и ГЛОНАСС. Временное разделение информационного обмена внутри группы осуществляется делением цикла $T_{\text{Ц}}$ на n временных окон. В каждом временном окне связываются между собой дуплексно различные пары БЛА. В течение цикла информация от каждого абонента к каждому абоненту попадет многократно и может быть мажоритарно сложена. Такая организация информационного обмена внутри группы БЛА обеспечивает высокую степень достоверности в условиях возможного возрастания уровня помех. Схемы информационных связей при централизованном управлении и централизованном управлении с одним ведущим могут быть использованы при сравнительно небольшом количестве БЛА в группе. При увеличении численности группы БЛА растет нагрузка на канал связи и на вычислительные средства устройства управления. Одним из выходов является применение иерархических стратегий управления, при которых группа БЛА делится на кластеры (подгруппы), каждый из которых

имеет своего ведущего, а ведущие кластеров управляются ЦУУ, расположенным на борту одного из БЛА (общего ведущего), либо на земле. Общий ведущий производит оценку общей ситуации и расчет оптимальных управлений и оптимальных траекторий движения. При этом информационный обмен внутри каждого кластера между ведущими кластеров в целях увеличения достоверности доставки необходимо организовать по кольцевому принципу, разделив соответственно временные интервалы обмена. При кластерной стратегии управления усложняется характер информационного обмена между БЛА, вследствие чего предъявляются серьезные требования к бортовой аппаратуре связи, особенно ведущих кластеров. Помехи и сбои в каналах связи крайне негативно отражаются на работе всего кластера. К тому же, выход из строя ведущего кластера или общего ведущего к серьезным проблемам.

Заключение

В работе показано, что система информационного обмена алгоритма адаптивного группой БЛА может быть рассмотрена как многоканальная система массового обслуживания с ограниченной очередью. Каждый канал представляет собой приемно-передающий модем (ППМ), пропускная способность которого определяется объемом передаваемого трафика поступающего от каждого БЛА группы БЛА-диспетчеру и определяемого непосредственно алгоритмом адаптивного управления. Это дает возможность рассчитать технические параметры ППМ. Предложены две схемы организации информационного обмена – с центральным устройством управления и схема централизованного управления с одним ведущим.

References

[1] M. Wang and J. N. Liu, “Fuzzy logic-based real-time robot navigation in unknown environment with dead ends.”, *Robotics and Autonomous Systems*, vol. 56, no. 7, pp. 625–643, 2008. DOI: 10.1016/j.robot.2007.10.002.

[2] M. M. Mohamad, M. W. Dunnigan, and N. K. Taylor, “Ant colony robot motion planning”, *EUROCON 2005: Intern. conf. on computer as a tool (Belgrade, Serbia, November 21-24, 2005)*, Proc. N.Y.: IEEE, vol. 1, pp. 213–216, 2005. DOI: 10.1109/EURCON.2005.1629898.

[3] H.-P. Huang and S.-Y. Chung, “Dynamic visibility graph for path planning.”, *IEEE-RSJ intern. conf. on intelligent robots and systems: IROS 2004 (Sendai, Japan, Sept. 28 - Oct. 2, 2004)*, Proc. N.Y.: IEEE, vol. 3, pp. 2813–2818, 2004. DOI: 10.1109/IROS.2004.1389835.

[4] J. O. Wallgrun, “Voronoi graph matching for robot localization and mapping”, *Transactions on computational science IX. B.:*, Springer, pp. 76–108, 2010. DOI: 10.1007/978-3-642-16007-3_4.

[5] S. M. LaValle, “Planning algorithms”, *Camb.; N.Y.: Camb. Univ. Press*, p. 826, 2006.

[6] K. Yang and S. Sukkarieh, “3D smooth path planning for a UAV in cluttered natural environments”, *IEEE/RSJ intern. conf. on intelligent robots and systems: IROS 2008 (Nice, France, Sept. 22-26, 2008)*, Proc. N.Y.: IEEE, pp. 794–800, 2008. DOI: 10.1109/IROS.2008.4650637.

[7] J. J. Kuffner and S. M. LaValle, “RRT-connect: An efficient approach to to single-query path planning”, *IEEE intern. conf. on robotics and automation: ICRA’2000 (San Francisco, CA, USA, April 24-28, 2000)*, Proc. N.Y.: IEEE, vol. 2, pp. 995–1001, 2000. DOI: 10.1109/ROBOT.2000.844730.

[8] N. H. Sleumer and N. Tschichold-Gurman, *Exact cell decomposition of arrangements used for path planning in robotics*. Zurich: Inst. of Theoretical Computer Science, 1999. DOI: 10.3929/ethz-a-006653440.

[9] L. De Filippis, G. Guglieri, and F. Quagliotti, “A minimum risk approach for path planning of UAVs”, *J. of Intelligent and Robotic Systems*, vol. 61, no. 1, pp. 203–219, 2011.

[10] L. De Filippis, G. Guglieri, and F. Quagliotti, “Path planning strategies for UAVs in 3D environments”, *J. of Intelligent and Robotic Systems*, vol. 65, no. 1, pp. 247–264, 2012.

[11] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed:algorithms based on Hamilton-Jacobi formulations”, *J. of Computational Physics*, vol. 79, no. 1, pp. 12–49, 1988.

[12] S. Avasker, A. Domoshnitsky, M. Kogan, O. Kupervasser, H. Kutomanov, Y. Rofsov, I. Volinsky, and R. Yavich, “A method for stabilization of drone flight controlled by autopilot with time delay”, *SN Applied Sciences*, vol. 2, no. 225, pp. 1–12, 2020.

[13] P. Eickhoff, *Osnovy identichnosti sistem upravleniya*. M.: Mir, 1975.

[14] A. Aleksandrov, *Optimalnye i adaptivnye sistemy*. M.: Vyshay shkola, 1989.

[15] C. Cao and N. Hovakimyan, “L1 adaptive controller for systems with unknown time-varying parameters and disturbances in the presence of non-zero trajectory initialization error”, *International Journal of Control*, vol. 81, no. 7, pp. 1147–1161, 2008.

[16] A. I. Gavrilov, C. M. Tu, and E. A. Budnikova, “Synthesis of Automatic Control System Quadrocopter”, *Collection of Scientific Tr. ”Management In Marine And Aerospace Systems” (UMAS–2014)*, pp. 621–624, 2014.

[17] J. Borenstein and Y. Koren, “The vector field histogram-fast obstacle avoidance for mobile robots”, *IEEE Trans. on Robotics and Automation.*, vol. 7, no. 3, pp. 278–288, 1991. DOI: 10.1109/70.88137.

[18] I. Ulrich and J. Borenstein, “VFH+: Reliable obstacle avoidance for fast mobile robots”, *IEEE intern. conf. on robotics and automation (Leuven, Belgium, May 20, 1998)*:, Proc. N.Y.: IEEE, vol. 2, pp. 1572–1577, 1998. DOI: 10.1109/ROBOT.1998.677362.

[19] E. P. Kubyshkin, L. N. Kazakov, and D. I. Sterin, “Mathematical modeling of flight reconfiguration of a unmanned aerial vehicles group”, *2018 Systems of Signal Synchronization, Generating and Processing in Telecommunications, SYNCHROINFO 2018*, Minsk: Institute of Electrical and Electronics Engineers Inc., 2018, DOI: 10.1109/SYNCHROINFO.2018.8457015.

[20] L. N. Kazakov, V. A. Botov, and E. P. Kubyshkin, “Adaptive Algorithm for Flight Restructuring of a Group of Unmanned Aerial Vehicles”, *2022 Systems of Signal Synchronization, Generating and Processing in Telecommunications*, SYNCHROINFO 2022 - Conference Proceedings, 2022. DOI: 10.1109/SYNCHROINFO55067.2022.9840955.

[21] V. F. Zhuravlev, “Fundamentals of theoretical mechanics”, *M.: Fizmatlit*, 2001.

[22] Y. I. Degtyarev, *Operations research*. M.: M.: Vyshay shkola, 1986.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 1, 2023
journal homepage: www.mais-journal.ru

DISCRETE MATHEMATICS IN RELATION TO COMPUTER SCIENCE

On Computational Constructions in Function Spaces

A. N. Morozov[1]

DOI: 10.18255/1818-1015-2023-1-28-38

[1]P. G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 41A35, 41A45, 65D25
Research article
Full text in Russian

Received February 6, 2023
After revision February 27, 2023
Accepted March 1, 2023

Numerical study of various processes leads to the need for clarification (extensions) of the limits of applicability of computational constructs and modeling tools. In this article, we study the differentiability in the space of Lebesgue integrable functions and the consistency of this concept with fundamental computational constructions such as Taylor expansion and finite differences is considered. The function f from $L_1[a;b]$ is called (k,L)-differentiable at the point x_0 from $(a;b)$, if there exists an algebraic polynomial P, of degree no higher than k, such that the integral over the segment from x_0 then x_0+h for $f-P$ there is $o(h^{k+1})$. Formulas are found for calculating coefficients of such P, representing the limit of the ratio of integral modifications of finite differences $\Delta_h^m(f,x)$ to h^m, $m=1,\cdots,k$. It turns out that if $f\in W_1^l[a;b]$, and $f^{(l)}$ is (k,L)-differentiable at the point x_0, then f is approximated by a Taylor polynomial up to $o\big((x-x_0)^{l+k}\big)$, and the expansion coefficients can be found in the above way. To study functions from L_1 on a set, a discrete "global" construction of a difference expression is used: based on the quotient $\Delta_h^m(f,\cdot)$ and h^m the sequence is built $\big\{\Lambda_n^m[f]\big\}$ of piecewise constant functions subordinate to partitions half-interval $[a;b)$ into n equal parts. It is shown that for a (k,L)-differentiable at the point x_0 function f the sequence $\big\{\Lambda_n^m[f]\big\}$, $m=1,\cdots,k$, converge as $n\to\infty$ at this point to the coefficients of the polynomial approximating the function at it. Using $\big\{\Lambda_n^k[f]\big\}$ the following theorem is established: *"f from $L_1[a;b]$ belongs to $C^k[a;b]\iff f$ is uniformly (k,L)-differentiable on $[a;b]$"*. A special place is occupied by the study of constructions corresponding to the case $m=0$. We consider them in $L_1[Q_0]$, where Q_0 is a cube in the space $\mathbb{R}^d$. Given a function $f\in L_1$ and a partition τ_n of a semi-closed cube Q_0 on n^d equal semi-closed cubes we construct a piecewise constant function $\Theta_n[f]$, defined as the integral average f on each cube $Q\in\tau_n$. This computational construction leads to the following theoretical facts: *1) if from L_1 belongs to L_p, $1\le p<\infty$, $\iff\big\{\Theta_n[f]\big\}$ converges in L_p; the boundedness of $\big\{\Theta_n[f]\big\}\iff f\in L_\infty$; 2) sequences $\big\{\Theta_n[\cdot]\big\}$ define on the equivalence classes the operator-projector Θ in the space L_1; 3) for the function $f\in L_\infty$ we get $\overline{\Theta[f]}\in B$, where B is the space of bounded functions, and $\overline{\Theta[f]}$ is the function $\Theta[f](x)$, extended on a set of measure zero and the equality $\big\|\overline{\Theta[f]}\big\|_B=\|f\|_\infty$*. Thus, in the family of spaces L_p one can replace $L_\infty[Q_0]$ with $B[Q_0]$.

Keywords: Difference Expressions; Integral Averaging; Continuity Operator; the Taylor Polynomial; Numerical Finding of Derivatives on a Computer

INFORMATION ABOUT THE AUTHORS

Anatoly Nikolaevich Morozov
correspondence author
orcid.org/0000-0001-9940-159X. E-mail: moroz@uniyar.ac.ru
PhD.

Funding: The work was carried out within the framework of the initiative research work of the YarGU. P. G. Demidov No. VIP-016.

For citation: A. N. Morozov, "On Computational Constructions in Function Spaces", *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 28-38, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 1, 2023
сайт журнала: www.mais-journal.ru

DISCRETE MATHEMATICS IN RELATION TO COMPUTER SCIENCE

О вычислительных конструкциях в функциональных пространствах

А. Н. Морозов[1]

DOI: 10.18255/1818-1015-2023-1-28-38

[1]Ярославский государственный университет им. П. Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 519.65
Научная статья
Полный текст на русском языке

Получена 6 февраля 2023 г.
После доработки 27 февраля 2023 г.
Принята к публикации 1 марта 2023 г.

Численное исследование различных процессов приводит к необходимости уточнения (расширения) границ применимости вычислительных конструкций и инструментов моделирования. В настоящей статье изучается дифференцируемость в пространстве интегрируемых по Лебегу функций и рассматривается согласованность этого понятия с основополагающими вычислительными построениями такими, как разложение Тейлора и конечные разности. Функцию f из $L_1[a;b]$ назовём (k,L)-дифференцируемой в точке x_0 из $(a;b)$, если существует алгебраический многочлен P, степени не выше k, такой, что интеграл по отрезку от x_0 до x_0+h для $f-P$ есть $o(h^{k+1})$. Найдены формулы для вычисления коэффициентов такого P, представляющие собой предел отношения интегральных модификаций конечных разностей $\Delta_h^m(f,x)$ к h^m, $m=1,\cdots,k$. Получается, что если $f\in W_1^l[a;b]$, и $f^{(l)}$ является (k,L)-дифференцируемой в точке x_0, то f приближается тейлоровским многочленом с точностью $o\big((x-x_0)^{l+k}\big)$, а коэффициенты разложения могут быть найдены указанным выше способом. Для исследования функций из L_1 на множестве применяется дискретная «глобальная» конструкция разностного выражения: на основе частного $\Delta_h^m(f,\cdot)$ и h^m строится последовательность $\{\Lambda_n^m[f]\}$ кусочно-постоянных функций, подчинённых разбиениям полуинтервала $[a;b)$ на n равных частей. Показано, что для (k,L)-дифференцируемой в точке x_0 функции f последовательности $\{\Lambda_n^m[f]\}$, $m=1,\cdots,k$, сходятся при $n\longrightarrow\infty$ в этой точке к коэффициентам приближающего в ней функцию многочлена. С помощью $\{\Lambda_n^k[f]\}$ устанавливается теорема: *«f из $L_1[a;b]$ принадлежит $C^k[a;b]\iff f$ равномерно (k,L)-дифференцируема на $[a;b]$»*. Отдельное место занимает изучение построений, соответствующих случаю $m=0$. Их рассматриваем в $L_1[Q_0]$, где Q_0 – куб в пространстве $\mathbb{R}^d$. По заданной функции $f\in L_1$ и разбиению τ_n полузамкнутого куба Q_0 на n^d равных полузамкнутых кубов построим кусочно-постоянную функцию $\Theta_n[f]$, определяемую как интегральное среднее f на каждом кубе $Q\in\tau_n$. Данная вычислительная конструкция приводит к следующим теоретическим фактам: *1) f из L_1 принадлежит L_p, $1\le p<\infty$, $\iff\{\Theta_n[f]\}$ сходится в L_p; ограниченность $\{\Theta_n[f]\}\iff f\in L_\infty$; 2) последовательности $\{\Theta_n[\cdot]\}$ определяют на классах эквивалентности оператор-проектор Θ в пространстве L_1; 3) для функции $f\in L_\infty$ получаем $\overline{\Theta[f]}\in B$, где B – это пространство ограниченных функций, а $\overline{\Theta[f]}$ – доопределённая на множестве меры ноль функция $\Theta[f](x)$, и выполняется равенство $\big\|\overline{\Theta[f]}\big\|_B=\|f\|_\infty$.* Таким образом, в семействе пространств L_p можно заменить $L_\infty[Q_0]$ на $B[Q_0]$.

Ключевые слова: разностные выражения; интегральные усреднения; оператор непрерывности; многочлен Тейлора; численное нахождение производных

ИНФОРМАЦИЯ ОБ АВТОРАХ

Анатолий Николаевич Морозов
автор для корреспонденции
orcid.org/0000-0001-9940-159X. E-mail: moroz@uniyar.ac.ru
канд. физ.-мат. наук, доцент.

Финансирование: Работа выполнена в рамках инициативной НИР ЯрГУ им. П. Г. Демидова № VIP-016.

Для цитирования: A. N. Morozov, “On Computational Constructions in Function Spaces”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 28-38, 2023.

О вычислительных свойствах (k, L)-дифференцируемости

Как обычно, $L_p[I]$ обозначает пространство действительных измеримых функций, интегрируемых в степени p $(1 \le p < \infty)$ по Лебегу на отрезке $I = [a; b]$,

$$\|f\|_{L_p[I]} = \left(\int\limits_I |f(x)|^p \, dx \right)^{\frac{1}{p}};$$

при $p = \infty$ в этой части рассматривается $B[I]$ – пространство измеримых ограниченных на отрезке I функций, –

$$\|f\|_{B[I]} = \sup_{x \in I} |f(x)|.$$

Введём для краткости записи семейство пространств

$$X_p[I] = \begin{cases} L_p[I] & \text{при } p < \infty, \\ B[I] & \text{при } p = \infty. \end{cases}$$

Когда неясность исключена, сокращаем обозначения до X_p и $\|f\|_p$. Длину I обозначаем $|I|$.
Также используются $C^k = C^k[I]$ – пространство k раз непрерывно дифференцируемых на отрезке I функций – и $(1 \le p < \infty)$
$W_p^k = W_p^k[I] = \left\{ f : f^{(k-1)} \text{ абсолютно непрерывна на отрезке } I, f^{(k)} \in L_p[I] \right\}$
с нормами $\|f\|_p + \|f^{(k)}\|_p$.

Численное исследование различных процессов приводит к необходимости уточнения (расширения) границ применимости вычислительных конструкций и инструментов моделирования. Эффективность такого подхода была продемонстрирована, например, в работе А. П. Кальдерона и А. Зигмунда ([1]), где были даны приложения обобщённого локального понятия производной к изучению локальных свойств решений дифференциальных уравнений $(1 \le p \le \infty)$. В одномерном случае их формулировка приводит к следующему определению.

Определение 1. *Функция $f \in X_p[I]$ называется (k, p)-дифференцируемой в точке $x_0 \in I$, если существует алгебраический многочлен P степени не больше k, для которого выполняется*

$$\|f - P\|_{X_p[J_{x_0,h}]} = o(h^{k+\frac{1}{p}}) \quad \text{при } h \longrightarrow 0, \text{ где } J_{x_0,h} = [x_0 - h; x_0 + h] \cap I.$$

Легко убедиться, что такой многочлен может быть только один, его часто называют тейлоровским. При k=1, p=∞ данное определение совпадает с классическим определением дифференцируемости. В статье автора [2] показано, что для k=1, p=1 определение остаётся содержательным, если уменьшить требования к взаимоотношению функции и приближающего её многочлена. Для произвольного $k \in \mathbb{N}$ формулировка приобретает следующий вид.

Определение 2. *Функцию $f \in L_1[I]$ назовём (k, L)-дифференцируемой в точке $x_0 \in I$, если существует алгебраический многочлен P степени не больше k, для которого выполняется*

$$\int\limits_{J_{x_0,h}} (f - P)(x) \, dx = o(h^{k+1}) \quad \text{при } h \longrightarrow 0, \text{ где } J_{x_0,h} = [x_0 - h; x_0 + h] \cap I.$$

Покажем единственность многочлена из Определения 2. Его удобно записать в виде $P(x) = c_0 + c_1 \cdot (x - x_0) + \cdots + c_k \cdot (x - x_0)^k$.

Рассмотрим для $h > 0$ и $x \in [a;\ b-h]$ «одностороннюю» функцию Стеклова: $S_h(f, x) \stackrel{\text{def}}{=} \frac{1}{h} \int\limits_x^{x+h} f(t) dt$.

Далее, для $x \in [a;\ b-(k+1) \cdot h]$ положим (см. [3])

$$\Delta_h^k(f, x) \stackrel{\text{def}}{=} \sum_{j=0}^{k} (-1)^{k-j} \cdot \binom{k}{j} \cdot S_h(f, x + jh). \tag{1}$$

Отметим, если $F(x) = \int\limits_a^x f(t)dt$, то

$$S_h(f,x) = \frac{\Delta_h^1(F,\ x)}{h}, \quad \text{а} \quad \Delta_h^m(f,x) = \Delta_h^m\Big(\frac{1}{h}\ \Delta_h^1(F),x\Big) = \frac{1}{h}\ \Delta_h^{m+1}(F,x),$$

где

$$\Delta_h^m(F,x) = \sum_{j=0}^{m}(-1)^{m-j}\cdot\binom{m}{j}\cdot F(x+jh), \quad m\in\mathbb{N},$$

– обычная m-я разность функции F в точке x.

Для $h<0$ в формуле (1) рассматриваются значения $x\in[a+(k+1)\cdot h;\ b]$.

Предложение 1. *Если* $f\in L_1[a;b]$, $a<x_0<b$ *и*

$$\int_{x_0}^{x_0+h}\big(f(x)-c_0-c_1\cdot(x-x_0)-\cdots-c_k\cdot(x-x_0)^k\big)\ dx = o(h^{k+1}) \ \ \textit{при}\ h\longrightarrow 0,$$

то

$$i)\quad c_0 = \lim_{h\to 0}\ S_h(f,x_0);$$

$$ii)\quad c_m = \frac{1}{m!}\cdot\lim_{h\to 0}\ \frac{\Delta_h^m(f,x_0)}{h^m}, \quad m=1,\cdots,k.$$

При $x_0=a$ *или* $x_0=b$ *рассматриваются соответствующие односторонние пределы.*

Доказательство. Из условия сразу получается, что

$$\int_{x_0}^{x_0+h}\big(f(x)-c_0\big)\ dx = o(h) \quad \text{при}\ h\longrightarrow 0,$$

откуда следует пункт *i*).

Пусть $F(x) = \int\limits_a^x f(t)dt$, тогда

$$F(x) = F(x_0) + c_0\cdot(x-x_0) + \cdots + \frac{c_m}{m+1}\cdot(x-x_0)^{m+1} + o\big((x-x_0)^{m+1}\big)$$

и

$$\frac{\Delta_h^m(f,x_0)}{h^m} = \frac{\Delta_h^{m+1}(F,x_0)}{h^{m+1}}, \quad m=1,\cdots,k.$$

Для конечных разностей хорошо известно соотношение (см., например, [4], с.159, или [5], с.54):

$$\Delta_h^{m+1}(u^j,x) = (m+1)!\cdot h^{m+1}\cdot\delta_{j,\ m+1}, \quad j=0,\ 1,\ \ldots,\ m+1,$$

где $\delta_{j,i}$ – символ Кронекера. Также имеет место простая оценка: если $g\in B[I]$, то

$$\|\Delta_h^{m+1}(g)\|_B \overset{\text{def}}{=} \|\Delta_h^{m+1}(g,\cdot)\|_{B[a;\ b-(m+1)h]} \le 2^{m+1}\cdot\|g\|_B\,.$$

Получаем

$$\lim_{h\to 0}\ \frac{\Delta_h^m(f,x_0)}{h^m} = \lim_{h\to 0}\ \frac{m!\cdot c_m\cdot h^{m+1}+\Delta_h^{m+1}\big(o\big((x-x_0)^{m+1}\big),x_0\big)}{h^{m+1}} = m!\cdot c_m.$$

□

Пример 1. *Пусть $k > 2$ и f_k для определённости – чётная функция, задаваемая при $x \geq 0$ формулой*

$$f_k(x) = \sum_{j=1}^{\infty} \chi_{j,k}(x),$$

где

$$\chi_{j,k}(x) = \begin{cases} 1, & \text{если } x \in I_{j,k} = \left[\frac{1}{2^j} - \frac{1}{2^{kj}};\ \frac{1}{2^j}\right], \\ 0, & \text{если } x \notin I_{j,k}. \end{cases}$$

Тогда

$$\frac{1}{2^k-1}\cdot|h|^k \leq \left|\int_0^h f_k(x)\,dx\right| \leq \frac{2^{2k}}{2^k-1}\cdot|h|^k \quad \text{при} \quad |h| \leq \frac{1}{2},$$

т. е. f_k является $(k-2, L)$-дифференцируемой в нуле.

Доказательство. Достаточно для положительных h рассмотреть оценку сверху. Если число $m \in \mathbb{N}$ таково, что $\frac{1}{2^{m+1}} < h \leq \frac{1}{2^m}$, то

$$\int_0^h f_k(x)\,dx \leq \sum_{j=m}^{\infty} \frac{1}{2^{kj}} = \frac{2^{2k}}{2^k-1}\cdot\left(\frac{1}{2^{m+1}}\right)^k.$$

□

Из чего вытекает утверждение.

Данный пример указывает на связь (k, L)-дифференцируемости с фрактальными свойствами, следующий факт объединяет это понятие с классической теорией разложений Тейлора.

Предложение 2. *Пусть $f \in W_1^l[I]$ такова, что функция $f^{(l)}$ является (k, L)-дифференцируемой в точке $x_0 \in I$, тогда справедливо разложение*

$$f(x) = f(x_0) + a_1\cdot(x-x_0) + \cdots + a_{l+k}\cdot(x-x_0)^{l+k} + o\big((x-x_0)^{l+k}\big),$$

коэффициенты которого могут быть найдены по формулам

$$a_m = \frac{1}{m!}\cdot\lim_{h\to 0}\frac{\Delta_h^m(f, x_0)}{h^m} = \frac{1}{m!}\cdot\lim_{h\to 0}\frac{\Delta_h^m(f, x_0)}{h^m}, \quad m = 1, \cdots, l+k.$$

В конечных точках отрезка I подразумеваются односторонние разложения и пределы.

Доказательство. По условию выполняется

$$\int_{x_0}^{x}\left(f^{(l)}(t) - c_0 - c_1\cdot(t-x_0) - \cdots - c_k\cdot(t-x_0)^k\right)\,dt = o\big((x-x_0)^{k+1}\big) \text{ при } x \longrightarrow x_0,$$

Это равносильно тому, что

$$f^{(l-1)}(x) = f^{(l-1)}(x_0) + c_0\cdot(x-x_0) + \cdots + \frac{c_k}{k+1}\cdot(x-x_0)^{k+1} + o\big((x-x_0)^{k+1}\big).$$

Проинтегрировав данное равенство $l-1$ раз, получим искомое разложение:

$$f(x) = f(x_0) + a_1\cdot(x-x_0) + \cdots + a_{l+k}\cdot(x-x_0)^{l+k} + o\big((x-x_0)^{l+k}\big),$$

где

$$a_m = \frac{f^{(m)}(x_0)}{m!}, \quad m = 1, \cdots, l-1, .$$

$$a_m = \frac{c_j}{(j+1)\ \cdots\ (j+n)}, \quad m = j+l, \quad j = 0, \cdots, k.$$

Ровно это, исходя из свойств $\Delta_h^m(f, x_0)$, и дают формулы для вычисления a_m в условии Предложения 2.

□

Отметим, что при $l{=}1$ очевидно наличие обратного соотношения в утверждении Предложения 2: *если* $f \in W_1^1[I]$ *и в точке* $x_0 \in I$ *справедливо разложение*

$$f(x) = a_0 + a_1\cdot(x{-}x_0) + \cdots + a_{k+1}\cdot(x{-}x_0)^{k+1} + o\big((x{-}x_0)^{k+1}\big) \quad \textit{при } x \longrightarrow x_0,$$

то функция f' *является* (k, L)*-дифференцируемой в точке* x_0.

Для исследования дальнейших свойств (k, L)-дифференцируемых функций применим дискретную «глобальную» конструкцию разностного выражения.

По заданной функции $f \in L_1[a; b]$ и разбиению $\tau_n = \{[x_{i-1}; x_i)\}_{i=1}^n$ полуинтервала $[a; b)$ на n равных полуинтервалов построим ступенчатую функцию, определяемую формулой

$$\Lambda_n^k[f](x) = \frac{\Delta_h^k(f,\ x_{i-1})}{h^k} \quad \text{при } x \in [x_{i-1}; x_i), \quad i = 1, \cdots, n,$$

(заключительный справа полуинтервал замыкаем), где $h = \frac{b-a}{n(k+1)}$ [3].

Имеем следующую модификацию Предложения 1.

Предложение 3. *Если функция* f *является* (k, L)*-дифференцируемой в точке* $x_0 \in I$:

$$\int_{x_0}^{x_0+h} \big(f(x) - c_0 - c_1\cdot(x{-}x_0) - \cdots - c_k\cdot(x{-}x_0)^k\big)\ dx = o(h^{k+1}) \quad \textit{при } h \longrightarrow 0,$$

то в этой точке последовательности $\big\{\Lambda_n^m[f]\big\}$ *для* $m = 1, \ldots, k$ *сходятся при* $n \longrightarrow \infty$ *к значениям* $m!\cdot c_m$.

Доказательство. Из (k, L)-дифференцируемости функции f в точке x_0 следует её (m, L)-дифференцируемость в этой точке для $m = 1, \ldots, k$, что запишем в виде:

$$F(x) = F(x_0) + c_0\cdot(x{-}x_0) + \cdots + \frac{c_m}{m{+}1}\cdot(x{-}x_0)^{m+1} + o\big((x{-}x_0)^{m+1}\big), \quad F(x) = \int_a^x f(t)dt.$$

Рассмотрим некоторое разбиение τ_n отрезка $[a; b]$. Пусть полуинтервал $J_i = [x_{i-1}; x_i)$, $1 \le i \le n$, /при $i = n$ – отрезок $[x_{n-1};\ x_n]$/ из τ_n содержит точку x_0. Подставляя

$$\frac{\Delta_h^m(f, x_{i-1})}{h^m} = \frac{\Delta_h^{m+1}(F, x_{i-1})}{h^{m+1}}, \quad i = 1, \cdots, k,$$

получаем

$$\Big|\Lambda_n^m[f](x_0) - m!\cdot c_m\Big| = \left|\frac{\Delta_h^{m+1}\left(\frac{c_m}{m+1}\cdot x^{m+1} + o\big((x-x_0)^{m+1}\big),\ x_{i-1}\right)}{h^{m+1}} - m!\cdot c_m\right| =$$

$$= \left|\frac{\Delta_h^{m+1}\left(o\big((x-x_0)^{m+1}\big),\ x_{i-1}\right)}{h^{m+1}}\right|.$$

Пусть $(m{+}1)\cdot u = \max\{x_0{-}x_{i-1},\ x_i{-}x_0\}$ /при этом будет выполняться $u \le h \le 2u$ /, тогда (см. Предложение 1)

$$\Delta_h^{m+1}\left(o\big((x-x_0)^{m+1}\big),\ x_{i-1}\right) = o(u^{m+1}).$$

Поскольку условие $n \longrightarrow \infty$ влечёт $h \longrightarrow 0$, то получаем нужное утверждение. □

Аналогичная модификация имеет место и для Предложения 2 (с таким же доказательством), причём, наряду с последовательностями $\big\{\Lambda_n^m[f]\big\}$ в этом случае к такому же результату приводят последовательности $\big\{\lambda_n^m[f]\big\}$, построенные на основе классических конечных разностей Δ_h^m.

Отметим, что согласно теоремам 3 из [3] и 4 из [2] выполняется:

если $f \in C^k$, *то* $\big\{\Lambda_n^k[f]\big\}$ *и* $\big\{\lambda_n^k[f]\big\}$ *сходятся равномерно к* $f^{(k)}$;
если $f \in W_p^k$, *то* $\big\{\Lambda_n^k[f]\big\}$ *и* $\big\{\lambda_n^k[f]\big\}$ *сходятся по норме пространства* L_p *к* $f^{(k)}$.

1. Исследование сходимости вычислительной конструкции $\Theta_n[\,\cdot\,]$

В этой части подробнее остановимся на особенностях локального в интегральном смысле отличия заданной функции от постоянной (см. Предложение 1).

Будем рассматривать в качестве множества-носителя замкнутый куб в $\mathbb{R}^d$

$$Q_0 = [a_1; b_1] \times \cdots \times [a_d; b_d], \quad \text{где} \quad b_i - a_i = b_j - a_j, \quad i, j \in \{1, \cdots, d\},$$

– с евклидовой метрикой и использовать функциональные пространства $L_p = L_p[Q_0]$, $1 \le p < \infty$, $B = B[Q_0]$, $C = C[Q_0]$, определяемые соответственно тому, как они описаны в начале статьи, а также $L_\infty = L_\infty[Q_0]$ – пространство измеримых существенно ограниченных на кубе функций,

$$\|f\|_{L_\infty[Q_0]} = \operatorname*{ess\,sup}_{x \in Q_0} |f(x)|, \quad \text{где } x = (x_1, \dots, x_d).$$

Для работы с подкубами удобно ввести следующие обозначения: если рёбра куба Q параллельны осям координат и имеют длину $2r$, точка x – его центр, то пишем также $Q(x, r)$; а меру куба (аналогично любого измеримого по Лебегу множества в $\mathbb{R}^d$) обозначаем как $\left|Q\right|$.

Пусть τ_n – разбиение полузамкнутого куба $Q_0 = [a_1; b_1) \times \cdots \times [a_d; b_d)$ на n^d равных полузамкнутых кубов /каждый полуинтервал $[a_i; b_i)$, $i = 1, \dots, d$, разбивается на n одинаковых полуинтервалов/. Куб Q_0 снова замкнём, а добавленные при этом точки распределим по соответствующим кубам из разбиения τ_n. Для полученного разбиения замкнутого куба оставим то же обозначение.

По заданной функции $f \in L_1$ и разбиению τ_n построим кусочно-постоянную функцию, определяемую формулой:

$$\Theta_n[f](x) = \frac{1}{\left|Q\right|} \int\limits_Q f(t)\,dt \quad \text{при } x \in Q$$

на каждом кубе $Q \in \tau_n$.

Нам понадобится понятие *точки Лебега* (Lebesgue point) интегрируемой функции (см., например, [6], с. 351). Применительно к обсуждаемому случаю его удобно описать так.

Определение 3. *Для $f \in L_1$ точка $x \in Q_0$ называется точкой Лебега, если*

$$\lim_{r \to 0} \frac{1}{\left|D(x, r)\right|} \int\limits_{D(x,r)} \left|f(t) - f(x)\right| dt = 0, \quad \textit{где} \quad D(x, r) = Q(x, r) \cap Q_0. \tag{2}$$

В определении точки Лебега вместо кубов $Q(x, r)$ могут быть использованы шары и множества некоторых других видов (см., [6], с. 352).

Ключевой результат, связанный с данным понятием, имеет вид (дифференциальная теорема Лебега):

для $f \in L_1$ почти все точки $x \in Q_0$ удовлетворяют условию (2).

В контексте данного параграфа приходим к следующим соотношениям.

Предложение 4. *Если $x \in Q_0$ – точка Лебега функции f, то последовательность $\{\Theta_n[f]\}$ при $n \to \infty$ сходится в этой точке к значению $f(x)$. Если для функции f условие* (2) *выполняется равномерно по кубу Q_0, то последовательность $\{\Theta_n[f]\}$ сходится равномерно на Q_0; из этого вытекает непрерывность f.*

Доказательство. Рассмотрим первое утверждение. Для $n \in \mathbb{N}$ построим разбиение τ_n куба Q_0 и зададим функцию $\Theta_n[f]$. Пусть куб $Q_i = Q(x_i, r_n)$, $1 \le i \le n^d$, из разбиения τ_n содержит точку x, в которой функция f удовлетворяет условию

$$\frac{1}{\left|Q(x,r)\right|}\int\limits_{Q(x,r)}\left|f(t)-f(x)\right|dt\longrightarrow 0\quad\text{при}\quad r\longrightarrow 0.$$

Тогда
$$\left|\Theta_n[f](x)-f(x)\right|=\left|\frac{1}{\left|Q(x_i,r_n)\right|}\int\limits_{Q(x_i,r_n)}\big(f(t)-f(x)\big)\,dt\right|\le\frac{1}{\left|Q(x_i,r_n)\right|}\int\limits_{Q(x_i,r_n)}\left|f(t)-f(x)\right|dt.$$

Очевидно, что если $x\in Q(x_i,r_n)$, то $Q(x_i,r_n)\subseteq Q(x,\,2{\cdot}r_n)$. Поэтому

$$\left|\Theta_n[f](x)-f(x)\right|\le\frac{1}{\left|Q(x,r_n)\right|}\int\limits_{Q(x,2r_n)}\left|f(t)-f(x)\right|dt\quad\longrightarrow 0\quad\text{при}\quad n\longrightarrow\infty.$$

Аналогичное рассуждение можно провести для граничных точек x куба Q_0.

Рассмотрим второе утверждение. При условии равномерного выполнения соотношения (2) для функции f на Q_0 получаем равномерную по кубу оценку для разности $\left|\Theta_n[f](x)-f(x)\right|$. Из равномерной сходимости последовательности кусочно-постоянных функций $\left\{\Theta_n[f]\right\}$, подчинённых разбиениям куба на равные подкубы, сразу вытекает непрерывность предельной функции (см. доказательство теоремы 3 данной статьи). Сам факт непрерывности функции, для которой равномерно на Q_0 выполняется (2) /в эквивалентной форме/ хорошо известен.

□

Отметим, что по Предложению 4 в качестве критериев принадлежности функции f из L_1 пространству C получаются: *выполнение условия* (2) *равномерно по кубу* Q_0 *или равномерная сходимость последовательности* $\left\{\Theta_n[f]\right\}$ *на* Q_0.

Приходим к следующему описанию пространств L_p.

Теорема 1. *Для того, чтобы функция f из L_1 принадлежала пространству L_p, $1\le p<\infty$, необходимо и достаточно, чтобы последовательность $\left\{\Theta_n[f]\right\}$ сходилась при $n\longrightarrow\infty$ в пространстве L_p. Равномерная ограниченность последовательности $\left\{\Theta_n[f]\right\}$ равносильна принадлежности функции f пространству L_∞.*

Доказательство. Пусть $f\in L_1$, и последовательность $\left\{\Theta_n[f]\right\}$ сходится в пространстве L_p. Из дифференциальной теоремы Лебега и Предложения 4 следует, что $\left\{\Theta_n[f]\right\}\underset{\text{п.в.}}{\longrightarrow}f$ (сходимость почти всюду). Таким образом, $f\in L_p$.

Рассмотрим обсуждаемое утверждение в обратную сторону. Пусть $f\in L_p$. Для заданного $\varepsilon>0$ найдётся функция $f_\varepsilon\in C$ такая, что $\left\|f-f_\varepsilon\right\|_p<\varepsilon$. Получим, что в неравенстве

$$\left\|\Theta_n[f]-f\right\|_p\le\left\|\Theta_n[f]-\Theta_n[f_\varepsilon]\right\|_p+\left\|\Theta_n[f_\varepsilon]-f_\varepsilon\right\|_p+\left\|f-f_\varepsilon\right\|_p$$

первое и третье слагаемые в правой части могут быть сделаны одновременно сколь угодно малыми за счёт выбора функции f_ε, а второе слагаемое станет меньше любой требуемой величины, начиная с некоторого достаточно большого номера n. Уточним оценку первого слагаемого.

$$\left\|\Theta_n[f]-\Theta_n[f_\varepsilon]\right\|_p=\left\|\Theta_n[f-f_\varepsilon]\right\|_p=\left(\sum_{i=1}^{n^d}\left|\frac{1}{|Q_i|}\int\limits_{Q_i}(f-f_\varepsilon)(t)\,dt\right|^p|Q_i|\right)^{\frac{1}{p}}\le$$

$$\le\left(\sum_{i=1}^{n^d}\int\limits_{Q_i}\left|(f-f_\varepsilon)(t)\right|^p dt\right)^{\frac{1}{p}}=\left\|f-f_\varepsilon\right\|_p.$$

Здесь на каждом кубе Q_i было применено интегральное неравенство Гёльдера с показателями $\frac{p}{p-1}$ и p.

Рассмотрим второе утверждение. Ограниченность последовательности $\{\Theta_n[f]\}$ вместе со сходимостью $\{\Theta_n[f]\} \underset{\text{п.в.}}{\longrightarrow} f$ дают $f \in L_\infty$. С другой стороны, ясно, что $\|\Theta_n[f]\|_\infty \le \|f\|_\infty$. □

На основе последовательностей $\{\Theta_n[\cdot]\}$ в рамках пространства L_1 получаем оператор:

$$\Theta[f] \stackrel{\text{def}}{=} \lim_{n\to\infty} \Theta_n[f].$$

Отметим простейшие свойства данного оператора:

1) если $\lim\limits_{r\to 0} \dfrac{1}{|D(x,r)|} \int\limits_{D(x,r)} |f(t)-y|\,dt = 0$, где $D(x,r)=Q(x,r)\cap Q_0$, то $\Theta[f](x) = y$;

2) $\Theta[\Theta[f]] = \Theta[f]$;

3) $\|\Theta_n[f]\|_1 \le \|f\|_1$, $\|\Theta\| = 1$.

Далее, для каждой непрерывной функции f выполняется $\Theta[f](x) \equiv f(x)$ (ровно для таких f последовательность $\{\Theta_n[f]\}$ сходится равномерно), и оператор, естественно, является ограниченным на C с интегральной метрикой. Из этого следует, что действие Θ в L_1 можно рассматривать как распространение его с пространства C /о единственности распространения непрерывного линейного оператора см., например, [7] с. 240/. Θ можно назвать ***оператором непрерывности***.

При $d = 1$ в Предложении 4 условие «x – точка Лебега» можно заменить на условие

$$f(x) = \lim_{h\to 0} \frac{1}{h} \int\limits_{x}^{x+h} f(t)\,dt, \quad x+h \in [a;b],$$

что равносильно дифференцируемости в точке x функции F – первообразной f. Выполнение данного соотношения в точке x для функции $f \in L_1[a;b]$ можно назвать ***L-непрерывностью в этой точке***.

Функция $\Theta[f]$ может быть неопределена на некотором фиксированном множестве меры ноль точек куба Q_0, в частности, – в точках разрыва кусочно-постоянной функции f. На основе поведения последовательности $\{\Theta_n[f]\}$ можно доопределить функцию $\Theta[f]$ на всём кубе Q_0, что даёт, в том числе, важные теоретические следствия.

Теорема 2. *Для каждой функции $f \in L_\infty$ имеет место $\overline{\Theta[f]} \in B$, где $\overline{\Theta[f]}$ – это доопределённая на множестве меры ноль усреднением частичных пределов последовательности $\{\Theta_n[f](x)\}$ функция $\Theta[f](x)$, и выполняется равенство $\|\overline{\Theta[f]}\|_B = \|f\|_\infty$.*

Доказательство. По Теореме 2 условие $f \in L_\infty$ равносильно ограниченности последовательности $\{\Theta_n[f]\}$. Более точно, $\|\Theta_n[f]\|_B \le \|f\|_{L_\infty}$ при всех $n \in \mathbb{N}$. Следовательно, для любой точки $x \in Q_0$ числовая последовательность $\{y_n\}$, где $y_n \stackrel{\text{def}}{=} \Theta_n[f](x)$, ограничена. Определим,

$$\overline{\Theta[f]}(x) = \frac{\liminf\{y_n\} + \limsup\{y_n\}}{2}.$$

Если в точке x выполняется условие (2), то $\overline{\Theta[f]}(x) = \Theta[f](x)$. Таким образом, $\overline{\Theta[f]}(x) = f(x)$ почти всюду и $\overline{\Theta[f]} \in B$.

□

Получается, что в рассматриваемой шкале лебеговых пространств «предельный» случай – пространство $L_\infty[Q_0]$ – можно заменить на пространство $B[Q_0]$. Этот факт имеет понятные обобщения, но их аккуратное доказательство уводит от темы статьи. Наметим только шаги и ориентиры. Для множества-носителя Π_0, где $\Pi_0 = [a_1; b_1] \times \cdots \times [a_d; b_d]$ – параллелепипед в $\mathbb{R}^d$, Предложение 4 доказывается аналогично случаю с кубом, поскольку в формуле (2) можно рассматривать семейство параллелепипедов Π, подобных Π_0 (см., [6], с. 352 Следствие 5.6.3), рассуждения Теорем 1 и 2 сохраняются полностью. Далее, все утверждения естественно справедливы для полузамкнутых и открытых параллелепипедов, поэтому Теорема 2 сразу переносится на каждое G_0, представляющее собой счётное объединение любых непересекающихся параллелепипедов, а в итоге приходим к измеримому по Лебегу множеству в $\mathbb{R}^d$.

2. Обратная теорема для (k, L)-дифференцируемости

Важной стороной введённых в первой части производных в вычислительных целях являются свойства функций, $(k; L)$-дифференцируемых на множестве. В то же время в теоретическом плане место таких производных определяется двусторонними соотношениями с классическими производными.

Определим обычные X_p-модули гладкости ($1 \le p \le \infty$):

$$\omega_k(f, t)_p = \sup_{0<h<t} \left\| \Delta_h^k(f) \right\|_{X_p[a;\, b-kh]}.$$

Как известно,

$$\| \cdot \|_p + \sup_{t>0} t^{-k} \omega_k(\cdot, t)_p$$

– норма на пространстве W_p^k, $1 < p < \infty$, (см. [8], теорема 2), и имеет место равенство:

$$\sup_{t>0} t^{-k} \omega_k(f, t)_p = \limsup_{h \to 0} \left\| h^{-k} \Delta_h^k(f) \right\|_{L_p[a;\, b-kh]} = \left\| f^{(k)} \right\|_p.$$

Теорема 3. *Для того, чтобы функция f из $L_1[I]$ принадлежала пространству $C^k[I]$ необходимо и достаточно, чтобы f была равномерно (k, L)-дифференцируемой на I.*

Доказательство. Любая функция f из $C^k[I]$ является, конечно, равномерно (k, L)-дифференцируемой на I. Это легко выводится, например, из хорошо известной формулы разложения k раз дифференцируемой функции с остаточным членом в форме Лагранжа: если на интервале между x_0 и x определена функция $f^{(k)}$, то

$$f(x) = f(x_0) + f'(x_0)\cdot(x-x_0) + \cdots + \frac{f^{(k-1)}(x_0)}{(k-1)!}\cdot(x-x_0)^{k-1} + \frac{f^{(k)}(\xi)}{k!}(x-x_0)^k,$$

где точка ξ заключена между x_0 и x. Последнее слагаемое можно записать в виде

$$\frac{f^{(k)}(x_0)}{k!}\cdot(x-x_0)^k + \alpha(x-x_0)\cdot(x-x_0)^k, \quad \text{где} \quad \alpha(x-x_0) = \frac{f^{(k)}(\xi) - f^{(k)}(x_0)}{k!}.$$

При условии непрерывности $f^{(k)}$ на I получаем, что все функции $\alpha(x - x_0)$ по модулю можно одновременно ограничить сколь угодно малым положительным числом ε для любых точек x_0 и x из I, удалённых друг от друга не больше, чем на соответствующее число δ. Из этого следует равномерная (k, L)-дифференцируемость функции f на I.

Рассмотрим достаточность. По условию равномерно по точкам x_0 из I выполняется

$$F(x) = F(x_0) + c_0(x_0)\cdot(x-x_0) + \cdots + \frac{c_k(x_0)}{k+1}\cdot(x-x_0)^{k+1} + o\big((x-x_0)^{k+1}\big) \quad \text{при } x \longrightarrow x_0,$$

где F – первообразная функции f.

Пусть $f_L^{(k)}(x_0) \stackrel{\text{def}}{=} k!\cdot c_k(x_0)$. Тогда равномерно на I есть соотношение (см. Предложение 1)

$$\frac{\Delta_h^{k+1}\big(F, x_0\big)}{h^{k+1}} = f_L^{(k)}(x_0) + \frac{\Delta_h^{k+1}\big(o\big((x-x_0)^{k+1}\big), x_0\big)}{h^{k+1}} = f_L^{(k)}(x_0) + o\big(h^0\big) \quad \text{при } h \longrightarrow 0. \tag{3}$$

Кроме того, $f_L^{(k)} \in C[I]$. Действительно, применяя рассуждения из доказательства Предложения 3, получаем равномерную сходимость на I последовательности $\big\{\mathbf{\Lambda}_n^k[f]\big\}$ к функции $f_L^{(k)}$. Поскольку разбиения отрезка на равные части обладают свойством «смешения узлов» (см., например, [5], с.241), то из равномерной сходимости последовательности ступенчатых функций, подчинённых таким разбиениям, следует непрерывность предельной функции (предположив противоположное, тот час же получим противоречие).

На основе условия $f_L^{(k)} \in C[I]$ получаем из (3), что

$$\sup_{t>0}\, t^{-(k+1)}\omega_{k+1}(F,t)_p = \limsup_{h\longrightarrow 0} \Big\| h^{-(k+1)}\cdot\Delta_h^{k+1}(F) \Big\|_{X_p[a;\, b-(k+1)h]} = \Big\|f_L^{(k)}\Big\|_p < \infty.$$

Это даёт принадлежность функции F пространствам W_p^{k+1}, следовательно, принадлежность f пространствам W_p^k при всех $1 < p < \infty$ и равенство $f^{(k)} = f_L^{(k)}$ в L_p.

Также можно было, доказав непрерывность функции $f_L^{(k)}$, завершить доказательство на основе соотношения (3) применением теоремы 1 из [8]. □

References

[1] A. P. Calderon and A. Zygmund, “Local Properties of Solution of Elliptic Partial Differential Equation”, *Studia Mathematica*, vol. 20, no. 2, pp. 171–225, 1961.

[2] A. N. Morozov, “Numerical Modeling Tools and S-Derivatives”, *Modeling and analysis of inform. systems*, vol. 29, no. 1, pp. 20–29, 2022.

[3] A. N. Morozov, “Calculation of Derivatives in the L_p Spaces where $1 \le p \le \infty$,”, *Modeling and analysis of inform. systems*, vol. 27, no. 1, pp. 124–131, 2020.

[4] V. K. Dzyadyk, *Introduction to the Theory of Uniform Approximation of Functions by Polynomials*. Nauka, 1977.

[5] L. L. Schumaker, *Spline Functions: Basic Theory*. Wiley, New York, 1981.

[6] V. I. Bogachev, *Measure Theory. V.1*. Springer, 2007.

[7] L. V. Kantorovich and G. P. Akilov, *Functional Analysis*. Nauka, Moscow, 1984.

[8] Y. A. Brudnyi, “Criteria for the Existence of Derivatives in Lp”, *Mathematics of the USSR-Sbornik*, vol. 2, no. 1, pp. 35–55, 1967.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 1, 2023
journal homepage: www.mais-journal.ru

THEORY OF COMPUTING

C Language Extension to Support Procedural-Parametric Polymorphism

A. I. Legalov[1], P. V. Kosov[1]

DOI: 10.18255/1818-1015-2023-1-40-62

[1]Higher school of Economics, National research University, 20, Myasnitskaya str., Moscow 101000, Russia.

MSC2020: 68N15, 68Q55
Research article
Full text in Russian

Received November 10, 2022
After revision February 3, 2023
Accepted February 8, 2023

Software development is often about expanding functionality. To improve reliability in this case, it is necessary to minimize the change in previously written code. For instrumental support of the evolutionary development of programs, a procedural-parametric programming paradigm was proposed, which made it possible to increase the capabilities of the procedural approach. This allows to extend both data and functions painlessly. The paper considers the inclusion of procedural-parametric programming in the C language. Additional syntactic constructions are proposed to support the proposed approach. These constructions include: parametric generalizations, specializations of generalizations, generalizing functions, specialization handlers. Their semantics, possibilities and features of technical implementation are described. To check the possibilities of using this approach, models of procedural-parametric constructions in the C programming language were built. The example in the article demonstrates the flexible extension of the program and support of multiple polymorphism.

Keywords: programming languages; compilation; procedural-parametric programming; polymorphism; multiple polymorphism; evolutionary software development

INFORMATION ABOUT THE AUTHORS

Alexander I. Legalov correspondence author	orcid.org/0000-0002-5487-0699. E-mail: alegalov@hse.ru Doctor of Technical Sciences, Professor.
Pavel V. Kosov	orcid.org/0000-0002-9035-312X. E-mail: pvkosov@hse.ru Graduate Student.

For citation: A. I. Legalov and P. V. Kosov, “C Language Extension to Support Procedural-Parametric Polymorphism”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 40-62, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 1, 2023
сайт журнала: www.mais-journal.ru

THEORY OF COMPUTING

Расширение языка C для поддержки процедурно-параметрического полиморфизма

А. И. Легалов[1], П. В. Косов[1]

DOI: 10.18255/1818-1015-2023-1-40-62

[1]Национальный исследовательский университет «Высшая школа экономики», ул. Мясницкая, д. 20, г. Москва, 101000 Россия.

УДК 004.4'42, 004.43
Научная статья
Полный текст на русском языке

Получена 10 ноября 2022 г.
После доработки 3 февраля 2023 г.
Принята к публикации 8 февраля 2023 г.

Разработка программного обеспечения зачастую связана с расширением функциональности. Для повышения надежности в этом случае необходимо минимизировать изменение ранее написанного кода. Для инструментальной поддержки эволюционной разработки программ была предложена процедурно-параметрическая парадигма программирования, что позволило повысить возможности процедурного подхода. Это обеспечивает безболезненное расширение как данных, так функций, используя при этом статическую типизацию. В работе рассматривается включение процедурно-параметрического программирования в язык С. Предлагаются дополнительные синтаксические конструкции, ориентированные на поддержку предлагаемого подхода. К ним относятся: параметрические обобщения, специализации обобщений, обобщающие функции, обработчики специализаций. Описываются их семантика, возможности и особенности технической реализации. Для проверки возможностей использования данного подхода построены модели процедурно-параметрических конструкций на языке программирования С. Приведенный пример демонстрирует гибкое расширение программы и поддержку множественного полиморфизма.

Ключевые слова: языки программирования; компиляция; процедурно-параметрическое программирование; полиморфизм; множественный полиморфизм; эволюционная разработка программного обеспечения

ИНФОРМАЦИЯ ОБ АВТОРАХ

Александр Иванович Легалов автор для корреспонденции	orcid.org/0000-0002-5487-0699. E-mail: alegalov@hse.ru доктор технических наук, профессор.
Павел Владимирович Косов	orcid.org/0000-0002-9035-312X. E-mail: pvkosov@hse.ru аспирант.

Для цитирования: A. I. Legalov and P. V. Kosov, "C Language Extension to Support Procedural-Parametric Polymorphism", *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 40-62, 2023.

Введение

При разработке программных систем необходимо учитывать различные дополнительные факторы, многие из которых определяются как критерии качества программного обеспечения (ПО). Расширение программы без изменения ранее написанного кода является одним из них. Оно обуславливается как неполным знанием окончательной функциональности программ во время ее создания и начальной эксплуатации, так и необходимостью ускорить выход продукта на рынок за счет реализации только базового набора функций с последующим его наращиванием. В подобных ситуациях добавление новых программных конструкций в код, сопровождаемое его изменением, зачастую ведет к появлению ошибок и непредсказуемому поведению.

Эволюционное расширение программ во многом связано с использованием динамического полиморфизма, который обеспечивает обработку ветвей программы во время ее выполнения, не используя при этом явной проверки альтернативных вариантов. Это отличает его от статического полиморфизма, при котором альтернативные вычисления выявляются во время компиляции. Изначально решения, связанные с реализацией динамического полиморфизма, были предложены для объектно-ориентированной (ОО) парадигмы. Например, в бестиповых и динамически типизированных ОО языках используется «утиная типизация». Ее идея заключается в том, что если любые, даже несвязанные объекты, имеют одинаковый по сигнатуре метод, они, каждый по своему, могут обработать обращение к нему. Это решение реализовано в языках программирования Smalltalk [1], Python [2] и многих других.

В ОО языках со статической типизацией ключевым решением стало совместное использование механизмов наследования и виртуализации. Наследование обеспечило идентичность интерфейсов родительского и дочернего классов, а виртуализация позволила подменять реализацию методов в дочерних классах. Этот подход реализован в языках ОО программирования C++ [3], Java [4], C# [5] и других.

Поддержку динамического полиморфизма, основанную на иных принципах, стали включать и в процедурные языки. В языке Go [6], реализован механизм интерфейсов, позволяющий использовать в качестве альтернативных обработчиков функции, связанные со структурами данных. Аналогичный механизм на основе типажей реализован в языке программирования Rust [7].

Инструментальная поддержка эволюционной разработки во многом способствовала популярности объектно-ориентированного программирования (ООП). В частности, были предложены паттерны проектирования [8], обеспечивающие гибкое расширение программы. Вместе с тем следует отметить, что безболезненное расширение программы в рамках ООП имеет определенные ограничения. Например, невозможно прямое расширение мультиметодов. Предлагаемые для этого решения [9—11] в основном ведут к использованию дополнительных конструктивов и алгоритмов, при которых, чаще всего, объектно-ориентированный стиль смешивается с процедурным. Аналогичные проблемы, связанные с реализацией мультиметодов, существуют также в языках программирования Go и Rust.

Это ограничение обусловлено тем, что мультиметод по сути является внешней функцией, поддерживающей множественный динамический полиморфизм над своими аргументами, в отличии от виртуального метода класса, интрефейсов Go и типажей Rust, которые поддерживают только одиночный полиморфизм над своими программными объектами, являясь по сути монометодами. Следует отметить, что множественный динамический полиморфизм был реализован в языке ОО программирования CLOS [12]. Однако эта реализация оказалась излишне громоздкой, что не привело к ее заимствованию в других ОО языках, в частности, в C++ [13].

Для инструментальной поддержки эволюционно расширяемого множественного полиморфизма была предложена процедурно-параметрическая парадигма программирования [14]. Это позволило повысить возможности процедурного подхода в области разработки ПО. Решение базируется

на параметрическом механизме формирования отношений между данными и обрабатывающими их процедурами, который может быть реализован различными способами [15]. Это обеспечивает безболезненное расширение как данных, так функций, используя при этом статическую типизацию. Для апробации идеи разработан язык O2M, расширяющий язык программирования Оберон-2 [16]. Проведенные на его основе эксперименты по написанию кода позволили проанализировать возможности процедурно-параметрического программирования (ППП). Язык включает все возможности Оберона-2, и позволяет использовать его модули. Данная реализация также показала, что подход может быть достаточно безболезненно интегрирован как в уже существующие языки процедурного и функционального программирования, так и использоваться при разработке новых языков. Также показано, что процедурно-параметрический полиморфизм обеспечивает более гибкое эволюционное расширение программ по сравнению с другими методами поддержки динамического полиморфизма [17].

Вместе с тем следует отметить, что процедурно-параметрическая парадигма не получила распространения. Во многом это обуславливается как отсутствием широкого практического использования языков семейства Оберон, так и тем, что основные идеи и технологии базируются на других языках программирования, получивших более широкое применение. Нельзя сказать, что процедурное программирование в настоящее время находится в полном упадке. Например, язык программирования C входит в пятерку наиболее популярных языков по различным рейтингам, что обуславливается его гибкостью и относительной простотой, а также использованием в предметных областях, где ОО подход не является эффективным. В связи с этим в работе рассматривается решение, направленное на включение в язык программирования C конструкций, поддерживающих ППП. Предлагаются дополнительные синтаксические конструкции, ориентированные на поддержку предлагаемого подхода. Описываются их семантика, возможности и особенности отображения в более простые конструкции, применяемые в архитектурных решениях уровня системы команд.

1. Основные концепции процедурно-параметрической парадигмы

При описании особенностей процедурно-параметрической парадигмы используются следующие понятия:

- основа специализации;
- специализация обобщения;
- параметрическое обобщение;
- экземпляр параметрического обобщения или специализированная переменная;
- обобщающая функция;
- обработчик параметрической специализации или специализированная функция;
- вызов параметрической процедуры.

1.1. Основа специализации

Под основой специализации понимается любой независимый базовый или составной тип имеющий имя. Практически любой тип может использоваться в качестве типа специализации обобщения. Также в качестве основы может использоваться и параметрическое обобщение, которое по сути тоже является составным типом. Понятие основы не накладывает никаких ограничений на самостоятельное использование этих типов в программе и вводится только для терминологической стыковки с другими вводимыми понятиями. Они, как и ранее, остаются обычными типами.

В качестве примеров структур, используемых далее в качестве основ специализаций, можно привести представления треугольника, прямоугольника и круга:

```
struct Triangle {int a, b, c;};   // Треугольник
struct Rectangle {int x, y;};     // Прямоугольник
struct Circle {int r;};           // Круг
```

1.2. Специализация обобщения

Под специализацией обобщения (или просто «специализацией») понимается любая из основ специализации, включенная в качестве составной части в параметрическое обобщение. Каждая специализация задает в обобщении один из альтернативных вариантов реализации. По своей сути специализация близка по семантике к альтернативному полю объединения языка программирования C. Помимо этого, при определенных условиях, в качестве специализаций в обобщение могут включаться и неименованные структурные типы.

1.3. Параметрическое обобщение

Параметрическое обобщение (или просто «обобщение») может содержать несколько специализаций. При этом обобщение может расширяться путем добавления новых специализаций в различных единицах компиляции, что и отличает его от объединения языка C. В предлагаемом расширении данного языка оно строится на основе модификации структуры и называется обобщенной структурой, синтаксис которой можно описать следующим образом:

```
ОбобщеннаяСтруктура = Структура
     "<" СписокСпециализаций ">" ["const"] |
     "<" [ШаблонПризнака] ">".
Шаблон признака = ":".
СписокСпециализаций = СписокСпециализацийПоТипу | СписокСпециализацийПоПризнаку.
СписокСпециализацийПоТипу = ИмяТипа ";" {ИмяТипа ";"}.
СписокСпециализацийПоПризнаку = Признак {"," Признак} ":" Тип ";"
                                {Признак {"," Признак} ":" Тип ";"}.
Признак = идентификатор.
Тип = ИмяТипа | ОписаниеНеименованногоТипа.
```

Как и при описании обычных структур, обобщенные структуры также можно использовать в описании `typedef`.

В отличие от обычных структур обобщение дополнительно содержит список специализаций. Уникальность при этом может определяться уникальностью типов, задающих различные специализации, или уникальностью признаков. Во втором случае допускается множество альтернатив имеющих одинаковый тип, а также использование в качестве типов неименованных структур, непосредственно определяемых в обобщении.

Если необходимо сформировать обобщенную структуру, которая в дальнейшем не должна изменяться, то для этого можно использовать квалификатор `const`, следующий непосредственно после описания специализаций. В качестве примера можно привести обобщенную структуру, задающую дни недели:

```
// Дни недели
struct WeekDay {int week_number;}
    <Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday: void;>
    const;
```

В примере, за счет использования типа `void`, осуществляется имитация перечисления. Различные признаки, предшествующие типу, позволяют задать только семь альтернатив, с каждой из которой можно впоследствии связать свой полиморфный обработчик специализации. При отсутствии квалификатора `const` можно сымитировать эволюционно расширяемый перечислимый тип. Общее поле в основной структуре задает номер недели в году.

Зачастую появление в программе новых специализаций может осуществляться не одновременно, а в процессе ее разработки. В таких случаях целесообразно иметь возможность расширения набора специализаций обобщения по мере необходимости. Пусть на первом этапе необходимо сформировать специализацию обобщения для треугольника. Это можно осуществить созданием следующей первоначальной обобщенной структуры:

```
struct Figure {}<struct Triangle;>;
```

Допускается использовать первую часть обобщенной структуры без задания полей, что указывает на отсутствие общих данных у всех формируемых обобщений.

Последующее расширение обобщения может осуществляться как путем одиночного, так и группового включения новых специализаций. Описание их подключения осуществляется в соответствии со следующими синтаксическими правилами:

```
ДобавлениеСпециализаций = ССылкаНаОбобщеннуюСтруктуру
                        "+" "<" СписокСпециализаций ">".
```

Тогда добавление прямоугольника и круга может выглядеть следующим образом:

```
struct Figure + <struct Rectangle; struct Circle;>;
```

Возможна ситуация, когда первоначальное обобщение создается пустым. В этом случае, при задании признаков специализаций типами, их список будет отсутствовать, а его наполнение начнется с добавлением специализаций. Для текущего примера первоначальное формирование специализации – треугольника может быть реализовано следующим образом:

```
// В обобщенной структуре отсутствуют специализации
struct Figure {}<>;
// Включение треугольника в обобщенную фигуру
struct Figure + <struct Triangle;>;
```

Использование идентификации посредством признаков специализаций позволяет многократно использовать один и тот же тип. Пусть структура прямоугольника используется для задания ромба через его диагонали, а для задания отрезка через его длину используем структуру, определяющую круг. Помимо этого определим дополнительные имена типов основ специализаций с применением `typedef`:

```
typedef struct Triangle {int a, b, c;} Triangle;   // Треугольник
typedef struct Rectangle {int x, y;} Rectangle;    // Прямоугольник
typedef struct Circle {int r;} Circle;             // Круг
```

При изначально пустом наборе специализаций обобщения и использовании для идентификации специализаций признаков шаблон будет выглядеть следующим образом:

```
typedef struct Figure2 {}<:> Figure2;
```

Дальнейшее расширение обобщения, можно описать следующим образом:

```
// Добавление прямоугольника, ромба, отрезка
Figure2 + <rect, rhomb: Rectangle; section: Circle;>;
// Добавление треугольника и круга
Figure2 + <trian: Triangle; circ: Circle;>;
```

На первом этапе обобщение Figure2 расширяется за счет прямоугольника, ромба и отрезка указанной длины. В следующим описании добавляются треугольник и круг. Расширения обобщения могу осуществляться в разных единицах компиляции, в каждой из которых будут видны только свои специализации.

1.4. Экземпляры параметрических обобщений

Экземпляры параметрических обобщений являются переменными, сформированными на основе специализаций (специализированными переменными). Для каждой из таких переменных задается тип специализации. В ходе их описания допускается начальная инициализация. Возможно формирование как скалярных специализированных переменных, так и массивов различной размерности. Помимо этого можно задавать описание указателей на специализированные переменные, а также создавать массивы таких указателей. Указатели на специализации при этом могут ссылаться только на соответствующие специализированные переменные в отличие от указателей на параметрические обобщения, которые могут указывать на любые специализации, сформированные от соответствующего обобщения.

```
ОписаниеСпециализированнойПеременнной =
    ССылкаНаОбобщеннуюСтруктуру "<" [ ИмяТипа | Признак ] ">"
                        СписокСпециализированныхПеременных.
СписокСпециализированныхПеременных =
    СпециализированнаяПеременная {"," СпециализированнаяПеременная}.
СпециализированнаяПеременная =
    {"*"} ИмяСпециализированнойПеременной {"[" Размерность "]"}
        ["=" Инициализатор].
ССылкаНаОбобщеннуюСтруктуру = struct ИмяОбобщенной структуры | ИмяTypedef.
```

Размерность массива задается аналогично описанию размерности для обычных переменных языка C. Инициализация определяется в зависимости от размерности и типа формируемой переменной.

Запрещается создание переменных непосредственно от обобщения. То есть переменных, имеющих только структурную часть при отсутствии части, определяемой ее специализацией. Однако при необходимости можно добавлять в описание обобщений специализаций с признаком типа void или, при использовании явных признаков специализации, формировать аналогичный по назначению признак с типом void. Например, empty:void. Указатели на обобщения могут ссылаться на любые специализации данного обобщения, что указывает на возможность разыменования. Соответствующие действия могут осуществляться в ходе начальной инициализации или выполнения операции присваивания.

```
ОписаниеУказателейНаОбобщения = ТипОбобщения
                        СписокОбобщенныхУказателей.
СписокОбобщенныхУказателей = ОбобщенныйУказатель {"," ОбобщенныйУказатель}.
ОбобщенныйУказатель =
    "*"{"*"} ИмяОбобщенногоУказателя {"[" Размерность "]"}["=" Инициализатор].
```

Как и любые переменные языка C, обобщенные и специализированные переменные могут быть глобальными, статическими, локальными или динамическими. Их значения могут использоваться в качестве фактических параметров при вызове функций.

В качестве примера можно привести следующие варианты создания специализированных переменных и обобщенных указателей:

```
struct Figure<struct Triangle> t1 = {}<{3,4,5}>, t2;
struct Figure<struct Circle> c1 = {}<{10}>, *pc1 = &c1, c[10];
Figure2<rect> r1, r2[3] = {{}<{1,2}>,{}<{3,4}>,{}<{5,6}>},
              r3 = <{7,4}>, pr1 = &r1;
struct Figure *pf1 = &t1, **ppf1 = &pf1;
struct WeekDay<Monday> monday = {35}<>; // Меняется только номер недели
```

К особенностям инициализации специализаций можно отнести указание описываемых значений внутри угловых скобок. При этом для структурных значений необходимо также указывать дополнительные фигурные скобки, которые не нужны, если специализация является скаляром.

1.4.1. Рекурсивное расширение специализаций

Построение новых специализаций может также осуществляться на основе обобщенной структуры, что позволяет формировать цепочки уточнений произвольной длины. При этом следует отметить, что подобное введение новых специализаций возможно только в том случае, если предшествующий тип является обобщенным. Это позволяет контролировать процесс добавления новых уточнений во время компиляции. Например, для формирования новой ступени обобщения T можно включить в него обобщение T0:

```
struct T {int x, y;}<:>;
struct T + <t0: _Bool; t1: struct{double r; char s;}>;

struct T0 {int z;}<:>;
struct T + <t2: T0;>;
```

Тип T0 можно будет уточнять, добавляя для этого к нему новые специализации, которые также могут содержать обобщения, обеспечивающие их дальнейшее уточнение:

```
struct T00 {int a;}<:>;
struct T0 + <t00: T00;>;
```

Использование данного приема позволяет выстраивать сложные зависимости между типами, воспринимая при этом различные специализации как уточнения одного и того же типа. Для приведенного примера могут быть сформированы следующие специализации:

- из T ⟶ T<t0>, или T<t1> или T<t2>,
- из T<t2> ⟶ T<t2<t00>> и т. д.

Допускается также рекурсивное подключение к существующим обобщениям других обобщений, включая и подключение самого себя. При необходимости это позволяет выстраивать длинные статические цепочки, формируемые во время компиляции программы. В качестве примера можно расширить тип T специализацией, построенной на основе этого же типа:

```
struct T + <t3: T>;
T += t3: T;
```

Появляется возможность выстраивать специализации, обеспечивающие поддержку возможностей, эквивалентных декорированию:

```
T<t3>, T<t3<t3>>, T<t3<t3<t3<t3<t2<t00>>>>>>, ...
```

1.4.2. Операции над специализированными переменными

Над специализированными переменными, а также над специализированными и обобщенными указателями возможны операции доступа к отдельным полям для чтения данных и присваивания. Допускается манипуляция над отдельными полями как структурной части, так и специализации. При этом для полей структурной части специализированной переменной в качестве разделителя выступает, как обычно, точка (.). Поле обобщенной части отделяется от обозначения переменной восклицательным знаком (!). Например:

```
t1!a = 5;
t2 = t1;
pf1 = pc1;
monday.week_number = 24; // Понедельник 24-й недели
struct T<t0> b = {}<1>;  // Инициализация специализации базового типа
b! = 0;                  // Изменение значения специализации базового типа
```

В случаях, когда поля специализаций являются структурами, обращение к ним осуществляется с указанием имени поля, перед которым ставится точка. Обращение к структурной части специализированной переменной осуществляется также как и к обычной структурной переменной.

Операции над указателями на обобщения включают проверку типа специализации. Это напрямую позволяет определить основу специализации и осуществить корректное явное приведение к нужному типу для организации доступа к полям. Операция реализуется как функция _Spec_is, возвращающая булевское значение:

```
ПроверкаТипаСпециализации =
    "_Spec_is" "(" УказательНаОбобщение, ИмяТипа | Признак ")".
```

В случае, если признак специализации соответствует проверяемому типу, функция возвращает значение, равное 1. В противном случае возвращается 0.

1.5. Обобщающие параметрические функции

Обобщающие функции обеспечивают поддержку процедурно-параметрического полиморфизма. Их сигнатуры определяют единый интерфейс для обработчиков параметрических специализаций. Каждая обобщающая функция имеет от одного до нескольких обобщенных формальных параметров. Они задаются в виде указателей на обобщения. При вызове такой функции в нее передаются ссылки на специализированные переменные, комбинации которых и определяют конкретный обработчик специализации. Обобщающая параметрическая функция описывается следующими синтаксическими правилами:

```
ОбобщающаяФункция = ТипВозвращаемогоПараметра ИмяФункции
    "<" СписокОбобщенныхПараметров ">" "(" [ СписокФормальныхПараметров ] ")"
        ТелоОбобщающейФункции.
СписокОбобщенныхПараметров = ОбобщенныйПараметр {"," ОбобщенныйПараметр }.
ТелоОбобщающейФункции = ПустоеТело | ОбработчикПоУмолчанию.
ПустоеТело = "=" "0".
ОбобщенныйПараметр = ТипОбобщения "*" ИмяОбобщения.
```

Например, обобщающая функция вывода на печать геометрической фигуры может быть представлена следующим образом:

```
void PrintFigure<struct Figure *f>() = 0;
```

Обобщенные параметры задаются в угловых скобках (в отличие от прочих параметров, которые, как обычно, располагаются в круглых скобках при их наличии). Отсутствие тела в данном случае говорит о том, что необходимо написать обработчики специализаций для всех конкретных фигур, представленных параметрическим обобщением. Помимо этого может быть задан обработчик по умолчанию, который вызывается в тех случаях, когда для подставляемой комбинации специализаций не будет реализован свой обработчик. Простейшим вариантов подобного обработчика в таком случае может служить вызов функции прерывания программы:

```
void PrintFigure<struct Figure *f>() {
    printf("Incorrect Argument!!!\n")
    exit(-1);
}
```

Предполагается также, что перед вызовами обобщающих функций компилятором будет автоматически генерироваться код, проверяющий указатели.

1.6. Обработчики параметрических специализаций

Обработчики параметрических специализаций (или специализированные функции) вызываются при обращении к обобщающей функции. Они обеспечивают непосредственную манипуляцию конкретными специализациями, включенными в состав параметрических обобщений. Синтаксис обработчиков:

```
СпециализированнаяФункция = ТипВозвращаемогоПараметра ИмяФункции
    "<" СписокСпециализаций ">" "(" СписокФормальных параметров ")"
        ТелоСпециализированнойФункции.
СписокСпециализаций = СпециализированныйПараметр
    {"," СпециализированныйПараметр }.
СпециализированныйПараметр = ТипСпециализации "*" ИмяСпециализации.
```

Выбор обработчика осуществляется в зависимости от значений параметрических аргументов, подставляемых вместо формальных параметров. Они могут располагаться в разных единицах компиляции, добавляясь, например, по мере создания очередной специализации, и для обобщающей функции печати фигуры выглядеть следующим образом:

```
// Вывод параметров прямоугольника
void PrintFigure<struct Figure<struct Rectangle> *r>() {
    printf("Rectangle: x = %d, y = %d", r->!x, r->!y);
}
// Вывод параметров треугольника
void PrintFigure<struct Figure<struct Triangle> *t>() {
    printf("Triangle: a = %d, b = %d, c = %d", (*t)!a, (*t)!b, (*t)!c);
}
// Вывод параметров круга
void PrintFigure<struct Figure<struct Circle> *r>() {
    printf("Circle: r = %d", c->!r);
}
```

Следует отметить, что восклицательный знак, определяющий доступ к полю обобщенной части, ставится после знака указателя.

1.7. Вызовы параметрических функций

Вызов функции по сути запускает обработчик для конкретной комбинации специализаций. Он отличается от вызова обычной функции только наличием списка дополнительных фактических параметров, указывающих на специализации, по которым автоматически осуществляется выбор обработчика, что и определяет динамический полиморфизм. Вызов параметрической функции имеет следующий синтаксис:

```
ВызовПараметрическойФункции = ИмяФункции
        "<" СписокУказателейНаСпециализации ">"
        "(" [СписокФактическихПараметров] ")".
```

В качестве примеров можно привести вызов функции печати фигуры для различных выше описанных специализированных переменных:

```
// печать параметров треугольника t1
PrintFigure<&t1>();
// печать параметров круга c1
PrintFigure<&c1>();
// печать параметров круга c1 через указатель pc1
PrintFigure<pc1>();
// печать параметров круга c[5]
PrintFigure<&c[5]>();
// печать параметров круга c[5] через смещение указателя
PrintFigure<c+5>();
// печать параметров треугольника t1 через обобщенный указатель pf1
PrintFigure<pf1>();
// печать параметров треугольника t1 через указатель на указатель *ppf1
PrintFigure<*ppf1>();
```

2. Эволюционная поддержка множественного полиморфизма

Основное достоинство процедурно-параметрической парадигмы проявляется при эволюционном расширении мультиметодов. В качестве примера рассмотрим создание мультиметода, проверяющего возможность размещения первой фигуры внутри второй. Пусть изначально имеется информация только о специализациях прямоугольника и треугольника, заданных с использованием признаков и описанием typedef. В этом случае обобщающая функция может иметь следующий вид:

```
// Обобщенная функция, требующая обязательного переопределения
// для всех специализаций
_Bool Multimethod<Figure2 *first, Figure2 *second>() = 0;
```

Для ее видимости в других единицах компиляции, используемых для определения обработчиков специализаций достаточно прототипа:

```
_Bool Multimethod<Figure2*, Figure2*>();
```

В одной или нескольких единицах компиляции можно сформировать четыре функции, осуществляющие обработку различных комбинаций специализаций. Каждый обработчик специализации описывает одну комбинацию параметров и определяет метод ее обработки:

```
// Прямоугольник разместится внутри прямоугольника
_Bool Multimethod<Figure2<rect> *r1, Figure2<rect> *r2>() {
    return ((r1->!x < r2->!x) && (r1->!y < r2->!y)) ||
           ((r1->!x < r2->!y) && (r1->!y> < r2->!x));
}
// Прямоугольник разместится внутри треугольника
_Bool Multimethod<Figure2<rect> *r1, Figure2<trian> *t2>() {...}
// Треугольник разместится внутри прямоугольника
_Bool Multimethod<Figure2<trian> *t1, Figure2<rect> *r2>() {...}
// Треугольник разместится внутри треугольника
_Bool Multimethod<Figure2<trian> *t1, Figure2<trian> *t2>() {...}
```

Более сложные алгоритмы заменены многоточием. Приведенный пример показывает, что специализации, тип которых известен во время компиляции, обрабатываются так же, как и обычные переменные эквивалентного типа.

Добавление в программу специализаций для новых геометрических фигур осуществляется без изменений существующих единиц компиляции. Например, появление круга приведет к реализации дополнительных обработчиков специализаций в новых файлах, подключаемых к проекту:

```
// Прямоугольник разместится внутри круга
_Bool Multimethod<Figure2<rect> r1*, Figure2<circ> *c2>() {
    return ((r1->!x*r1->!x + r1->!y*r1->!y) < (c2->!r*c2->!r));
}
// Треугольник разместится внутри круга
_Bool Multimethod<Figure2<trian> *t1, Figure2<circ> *c2>() {...}
// Круг разместится внутри прямоугольника
_Bool Multimethod<Figure2<circ> *c1, Figure2<rect> *r2>() {...}
// Круг разместится внутри треугольника
_Bool Multimethod<Figure2<circ> *c1, Figure2<trian> *t2>() {...}
// Круг разместится внутри круга
_Bool Multimethod<Figure2<circ> *c1, Figure2<circ> *c2>() {
    return c1->!r < c2->!r;
}
```

3. Отображение процедурно-параметрических конструкций в низкоуровневое представление

Перед реализацией генератора кода, обеспечивающего трансформацию процедурно-параметрических конструкций в машинный код необходимо провести анализ возможных вариантов окончательно формируемого представления. Для этого можно непосредственно использовать язык программирования C, программные объекты которого практически однозначно отображаются на компьютерную память. Помимо этого использование данного языка позволяет гораздо проще оценить эффективность различных вариантов отображения. Основной идеей при этом, как и в объектно-ориентированных системах, является создание программных объектов и связей между ними, обеспечивающих поддержку новой парадигмы, до запуска функции `main`. Это обеспечивается за счет возможностей операционной системы Linux, использовать аналоги конструкторов, выполняемых до запуска `main`, и деструкторов, запускаемых после ее завершения [18].

3.1. Трансформация обобщений

Параметрические обобщения преобразуются в структуры, содержащие общие для всех специализаций поля. Помимо этого они содержат дополнительное поле, в котором фиксируется признак специализации. Признак задается числом от нуля до количества специализаций, окончательное число которых может формироваться различным образом: либо на этапе сборки программы, либо во время ее запуска, но до выполнения функции `main`. Например, описанное выше обобщение, задающее геометрическую фигуру, можно трансформировать в следующие программные объекты языка C:

```
typedef struct figure {
  unsigned tag;       // тег, задающий признак специализации
  struct {} head;     // первичная структура, возможен список полей
} figure;
```

Помимо этого для каждого обобщения создается статическая переменная, в которой фиксируется количество специализаций, сформированных в различных единицах компиляции:

```
static unsigned _figure_number_ = 0;
```

Для получения ее значения и изменения при регистрации новых специализаций порождаются функции, через которые и осуществляется доступ:

```
// Регистрация очередной специализации
void _figure_spec_register(void (*set_spec_tag)(unsigned));
// Получение текущего числа специализаций
unsigned get_figure_number();
// Возврат признака специализации через указатель на обобщение
unsigned get_figure_tag(figure *fig);
```

Прототипы этих функций определены в заголовочном файле вместе с описанием структуры обобщения и подключаются к единицам компиляции, отвечающим за установку параметров специализаций. Переменная `_figure_number_` и определения выше объявленных функций находятся в отдельной единице компиляции:

```
// Регистрация очередной специализации фигуры
void _figure_spec_register(void (*set_spec_tag)(unsigned)) {
    // Текущее значение становится тегом специализации
    set_spec_tag(_figure_number_);
    ++_figure_number_;  // изменение числа зарегистрированных фигур
}
// Возврат текущего количества зарегистрированных специализаций
unsigned get_figure_number() {
    return _figure_number_;
}
// Возврат признака специализации через указатель на обобщение
unsigned get_figure_tag(figure *fig) {
    return fig->tag;
}
```

Передаваемая через указатель `set_spec_tag` функция используется для получения каждой разновидности специализации своего тега. Эта функция непосредственно связана с переменной, идентифицирующей конкретную специализацию. После регистрации очередной специализации их общее количество увеличивается на единицу.

3.2. Трансформация специализаций

Специализации обобщений формируются по одному и тому же подходу. Описание каждой специализации и ее реализация находятся в заголовочном файле и в файле реализации. В качестве примера рассмотрим специализацию фигуры как прямоугольника. Пусть данная специализация использует в качестве основы ранее описанный прямоугольник:

```
struct rectangle {int x, y;};   // основа специализации
```

Будем считать, что при создании специализации обобщения формируется уникальное имя, не совпадающее с именами конструкций, определяемых программистом. Формируемый специализированный прямоугольник может напрямую не использовать исходную фигуру, повторяя при этом ее поля:

```
typedef struct figure_rectangle {
    // Повторение полей обобщения
    unsigned tag;   // поле тега на том же месте
    struct {} head; // поле структуры, идентичное структуре обобщения
    struct rectangle tail; // поле, определяющее содержание специализации
} figure_rectangle;
```

Помимо структуры задаются прототипы функций, используемые данной специализацией:

```
// Установщик признака специализации, передаваемый регистратору
void set_figure_rectangle_tag(unsigned tag);
// Получение признака специализации. Необходим при регистрации обработчиков
unsigned get_figure_rectangle_tag();
// Инициализация признака спецпеременной перед использованием
void init_figure_rectangle(figure_rectangle *p_fr);
```

Реализация этих функций и статической переменной `_figure_rectangle_tag`, хранящей признак данной специализации, размещаются в отдельной единице компиляции:

```
// Признак специализации. Доступен через интерфейсные функции
static unsigned _figure_rectangle_tag = -1;

void set_figure_rectangle_tag(unsigned tag) {
    _figure_rectangle_tag = tag;
}
unsigned get_figure_rectangle_tag() {
    return _figure_rectangle_tag;
}
void init_figure_rectangle(figure_rectangle *p_fr) {
    p_fr->tag = _figure_rectangle_tag;
}
```

В этой же единице компиляции определяется функция `register_figure_rectangle_tag`, осуществляющая регистрацию данной специализации. Именно она формирует значение признака до запуска функции `main`, используя специфику запуска программы операционной системой:

```
// Конструктор, обеспечивающий создание признака специализации
void  __attribute__ ((constructor(101))) register_figure_rectangle_tag() {
    _figure_spec_register(set_figure_rectangle_tag);
}
```

Число 101 определяет приоритет. Он одинаков для всех конструкторов данного типа, осуществляющих регистрацию специализаций, так как порядок их регистрации не играет значения. Основным ограничением является то, что приоритет должен быть больше 100, так как меньшие значения зарезервированы за операционной системой.

3.3. Трансформация обобщающих функций

Обобщающие функции предоставляют общий интерфейс для соответствующих обработчиков специализаций, предоставляя тем самым общую основу для динамического полиморфизма. Помимо этого они используются для формирования действий, связанных с обработкой по умолчанию. Последнее осуществляется в том случае, если обработчик специализации отсутствует. Основная идея обобщающей функции заключается в вызове обработчика специализации в соответствии с комбинацией специализаций, используемых в качестве фактических параметров. При этом идентификация нужного обработчика определяется по признакам специализаций. Возможны различные варианты реализации обобщающих функций и их подмены обработчиками специализаций.

1. Обобщающая функция вызывает нужные действия, обращаясь к обработчикам специализаций через массив указателей на соответствующие функции. Данный подход обеспечивает эффективность, эквивалентную скорости доступа к виртуальным методам при объектно-ориентированном подходе. Но в тех случаях, когда приоритетным является использование обработчика по умолчанию, хранение указателей в массиве может оказаться неэффективным.
2. Выбор нужной специализации может осуществляться через обход списка указателей с поиском нужных комбинаций признаков, например, используя для ускорения хеширование или древовидные схемы. Однако в данном случае скорость поиска нужной комбинации сильно зависит от числа обработчиков специализаций.
3. Возможна централизованная реализация с применением условных операторов или переключателей. Данный подход по сути является решением, типичным для процедурного программирования. Несмотря на свою простоту он требует знания всех специализаций и не поддерживает эволюционного расширения.

В качестве примера рассмотрим реализацию обобщающей функции вывода параметров геометрических фигур на основе массива указателей. Объявление обобщающей функции `print_figure` представлено в заголовочном файле:

```
// Одномерный массив указателей на обработчики специализаций
typedef void (*print_figure_pointer)(figure* p_f);
// Обобщающая функция
void print_figure(figure* p_f);
// Регистрация обработчика специализации в массиве обработчиков
void _print_figure_spec_register(
        unsigned (*get_spec_tag)(), print_figure_pointer spec);
```

Там же, для удобства использования, через объявление `typedef` приведено описание указателя `print_figure_pointer` на функции — обработчики специализаций, параметры которых совпадают с параметрами обобщающей функции. Помимо этого объявлен прототип функции, осуществляющей регистрацию обработчиков специализаций.

Описание указателя на массив обработчиков специализаций и определения приведенных выше функций представлено в отдельной единице компиляции. Указатель на массив обработчиков специализаций размещен в статической памяти и недоступен из других модулей программы:

```
static print_figure_pointer *_print_figure_array;
```

Доступ к нему осуществляется только через соответствующие функции, расположенные в этой же единице компиляции.

Сами массивы указателей создаются в конструкторах регистраторов обобщающих функций, формирующих массивы указателей на обработчики специализаций. Они имеют меньший приоритет (в данном случае он равен 201) по сравнению с конструкторами, регистрирующими специализации. Это позволяет создавать массивы, размерность которых соответствует количеству зарегистрированных специализаций. Конструктор, формирующий массив указателей на обработчики функций печати фигур реализован следующим образом:

```
void  __attribute__ ((constructor(201))) register_print_figure_array() {
    unsigned figure_number = get_figure_number();
    _print_figure_array =
        malloc(figure_number * sizeof(print_figure_pointer));
    for(unsigned i = 0; i < figure_number; ++i) {
        _print_figure_array[i] = print_figure_default;
    }
}
```

Получив количество зарегистрированных специализированных фигур, он формирует массив соответствующей размерности, заполняя его указателями на обработчик по умолчанию. Это позволяет осуществить корректный вызов обобщенной функции, если для каких-либо специализаций обработчик будет отсутствовать. Представленный ниже обработчик по умолчанию выводит сообщение об отсутствии функции вывода и завершает программу:

```
static void print_figure_default(figure* p_f) {
    printf("ERROR: print for figure is absent!\n");
    exit(1);
}
```

Выделение динамической памяти должно сопровождаться последующим ее освобождением по окончании выполнения функции `main` или выхода по функции `exit`. Для этого используется соответствующий деструктор:

```
void  __attribute__ ((destructor(201))) delete_print_figure_array() {
    free(_print_figure_array);
}
```

Регистрация обработчиков специализаций во многом аналогична регистрации специализаций. Имеется общий регистратор, которому передается указатель на функцию-обработчик:

```
void _print_figure_spec_register(
  unsigned (*get_spec_tag)(), print_figure_pointer spec) {
  // Передаваемый указатель на функцию фиксируется в массиве указателей
  unsigned tag = get_spec_tag();
  _print_figure_array[tag] = spec;
}
```

Запуск этого регистратора осуществляется конструктором специализации и рассмотрен ниже.

Обобщающая функция может вызвать любой из обработчиков специализаций. Она обращается к массиву обработчиков, используя в качества индексов теги специализаций, выступающих в роли обобщенных параметров. Для функции печати обобщенной фигуры функция выглядит следующим образом:

```
void print_figure(figure* p_f) {
    print_figure_pointer spec = _print_figure_array[p_f->tag];
    spec(p_f);
}
```

3.4. Трансформация специализаций

Приведенная схема позволяет полностью отделить обобщение от специализаций и их обработчиков, которые могут разрабатываться и эволюционно добавляться в отдельных единицах компиляции по одной и той же схеме. В качестве примера рассмотрим добавление в программу специализированного прямоугольника. Его основой служит структура, описывающая стороны прямоугольника:

```
struct rectangle {int x, y;};
```

Используя эту основу, в заголовочном файле можно описать специализацию:

```
typedef struct figure_rectangle {
  // Повторение полей обобщения
  unsigned tag;       // поле тега на том же месте
  struct {} head;     // поле структуры, идентичное структуре обобщения
  struct rectangle tail;  // поле, определяющее содержание специализации
} figure_rectangle;
```

В начале структуры, описывающей специализацию, повторяются поля, соответствующие обобщению. Это обеспечивает семантическую эквивалентность и возможность приведения к типу обобщенной фигуры при проведении различных манипуляций. После этого следуют данные, определяющие специализацию обобщения. В примере это структура, описывающая прямоугольник. Помимо этого в заголовочном файле описываются функции, связанные с обработчиком специализации:

```
// Установщик признака специализации
void set_figure_rectangle_tag(unsigned tag);
// Получение признака специализации. Необходимо в процессе регистрации фигуры
unsigned get_figure_rectangle_tag();
// Инициализация признака специализации перед использованием
void init_figure_rectangle(figure_rectangle *p_fr);
```

Эти функции определены в единице компиляции специализации.

Установщик признака передается на регистрацию специализации, которая выполняется с использованием функции `_figure_spec_register`.

```
void set_figure_rectangle_tag(unsigned tag) {
    _figure_rectangle_tag = tag;
}
```

Полученное в результате значение, являющееся общим для всех специализаций данного типа, сохраняется в статической переменной:

```
static unsigned _figure_rectangle_tag;
```

Доступ к значению переменной, для использования его в качестве признака, осуществляется посредством функции:

```
unsigned get_figure_rectangle_tag() {
    return _figure_rectangle_tag;
}
```

Помимо этого переменная используется для установки признака сформированного прямоугольника после его объявления в программе. Эта установка осуществляется функцией инициализации прямоугольника:

```
void init_figure_rectangle(figure_rectangle *p_fr) {
    p_fr->tag = _figure_rectangle_tag;
}
```

Сам процесс инициализации осуществляется конструктором, который, для корректного формирования количества специализаций, выполняется с наибольшим приоритетом:

```
void  __attribute__ ((constructor(101))) register_figure_rectangle_tag() {
    _figure_spec_register(set_figure_rectangle_tag);
}
```

3.5. Трансформация обработчиков специализаций

Добавление обработчиков специализаций связано с реализацией конкретных функций и их регистрацией в массивах обработчиков специализаций. В рассматриваемом примере функция вывода специализированного прямоугольника выглядит следующим образом:

```
static void print_figure_rectangle(figure* p_f) {
    figure_rectangle* p_fr = (figure_rectangle*)p_f;
    printf("rectangle: x= %d, y = %d\n", p_fr->tail.x, p_fr->tail.y);
}
```

В данном случае идет прямое обращение к специализации.

Для регистрации этой функции в массиве обработчиков обобщающей функции вывода используется соответствующий конструктор:

```
void  __attribute__ ((constructor(301))) register_print_figure_rectangle() {
    _print_figure_spec_register(get_figure_rectangle_tag,
                                print_figure_rectangle);
}
```

Для запуска после формирования массива указателей на специализации его приоритет задается ниже ранее описанных конструкторов.

3.6. Формирование процедурно-параметрических переменных

Специализированные переменные порождаются на основе абстракций, описывающих специализации. Их создание практически ничем не отличается от создания в языке C структурных переменных. Аналогичным образом для них осуществляется инициализация полей как общей, так и специализированной частей. Ключевым отличием является необходимость предварительной инициализации, при которой формируется признак специализации.

Инициализация внешних (глобальных и статических) переменных должна осуществляться до запуска функции `main`. Поэтому вызов соответствующей функции должен проходить внутри конструктора. Например, формирование переменной, описывающей прямоугольник можно представить следующим образом:

```
figure_rectangle r = {.tail.x=10, .tail.y=20};
// Конструктор, обеспечивающий инициализацию тега переменной r
void  __attribute__ ((constructor(401))) init_figure_rectangle_r() {
    init_figure_rectangle(&r);
}
```

Инициализация локальных переменных, а также переменных, размещаемых в динамической памяти осуществляется уже после запуска функции main. Например:

```
figure_rectangle r2 = {.tail.x=1, .tail.y=2};
init_figure_rectangle(&r2);

figure *p_f = (figure*)malloc(sizeof(figure_rectangle));
init_figure_rectangle((figure_rectangle*)p_f);
((figure_rectangle*)p_f)->tail.x = 100;
((figure_rectangle*)p_f)->tail.y = 200;
...
free(p_f);
```

3.7. Использование обобщающих функций

Процедурно-параметрический полиморфизм поддерживается использованием обработчиков специализаций, доступ к которым осуществляется через вызовы обобщающих функций. Каждая обобщающая функция обращается к специализациям через указатель на обобщение, запуская выбор нужного обработчика через признак специализации. При этом сам вызов нужного обработчика остается прозрачным и для представленных выше переменных может выглядеть следующим образом:

```
figure *p_f1 = (figure*)&r;
print_figure(p_f1); // Печать внешней специализированной переменной
p_f1 = (figure*)&r2;
print_figure(p_f1); // Печать локальной специализированной переменной
print_figure(p_f); // Печать переменной в динамически выделенной памяти
```

3.8. Варианты дальнейшего эволюционного расширения

Предлагаемый подход позволяет гибко наращивать функциональность программы, добавляя в нее как специализации, так и их обработчики без изменения ранее написанного кода. Например, добавление специализированного треугольника может формироваться в отдельных заголовочных файлах и соответствующих единицах компиляции по тому же принципу, что и формирование специализированного прямоугольника. Точно также в отдельной единице компиляции может быть сформирован обработчик, осуществляющий вывод специализированного треугольника.

Аналогичным образом осуществляется поддержка множественного полиморфизма. Допускается формирование обобщающих функций, аргументами которых являются два и более обобщения. Отличительной чертой в данном случае является то, что для регистрации обработчиков специализаций необходимо использовать многомерные массивы, размерность которых определяется количеством обобщенных аргументов функции. Вместе с тем их можно смоделировать через одномерные массивы с использованием для доступа формул, осуществляющих необходимый пересчет.

В качестве примера можно рассмотреть мультиметод, осуществляющий анализ вложенности первой геометрической фигуры во вторую. В заголовочном файле, как и ранее, описываются прототип обобщающей функции, осуществляющей запуск мультиметода, а также прототип регистратора

обработчиков специализаций. При этом регистратор размещает в массиве обработчики с использованием признаков первого и второго аргументов:

```
typedef void (*multimethod_figure_pointer)(figure* p_f1, figure* p_f2);
void multimethod_figure(figure* p_f1, figure* p_f2);
void _multimethod_figure_spec_register(unsigned (*get_spec_tag1)(),
                                       unsigned (*get_spec_tag2)(),
                                       multimethod_figure_pointer spec);
```

Эти функции определяются в соответствующей единице компиляции.

Помимо этого формируется одномерный массив, имитирующий матрицу обработчиков специализаций для двух аргументов. Размерность этого массива, а также формула, осуществляющая пересчет двумерной размерности в одномерную непосредственно прописаны в соответствующих функциях. Конструктор и деструктор реализуются аналогично тому, как это было сформировано для обобщающей функции вывода.

```
// Одномерный параметрический массив, имитирующий матрицу
static multimethod_figure_pointer *_multimethod_figure_array;
// Обобщающая функция, вызывающая обработчики специализаций
void multimethod_figure(figure* p_f1, figure* p_f2) {
  multimethod_figure_pointer spec =
    _multimethod_figure_array[get_figure_number()*p_f1->tag + p_f2->tag];
  spec(p_f1, p_f2);
}
// Обработчик по умолчанию, когда обработчик специализации отсутствует.
static void multimethod_figure_default(figure* p_f1, figure* p_f2) {
  printf("WARNING: Multimethod not implemented for this combination!\n");
}
// Конструктор, создающий матрицу указателей на обработчики специализаций
void
__attribute__ ((constructor(201))) register_multimethod_figure_array() {
  unsigned figure_number = get_figure_number();
  // Формируется матрица
  unsigned multimethod_array_size = figure_number * figure_number;
  _multimethod_figure_array =
    malloc(multimethod_array_size * sizeof(multimethod_figure_pointer));
  for(unsigned i = 0; i < multimethod_array_size; ++i) {
    _multimethod_figure_array[i] = multimethod_figure_default;
  }
}
// Деструктор, обеспечивающий освобождение памяти
void
__attribute__ ((destructor(201))) delete_multimethod_figure_array() {
  free(_multimethod_figure_array);
}
// Регистрация обработчика специализации в матрице
void _multimethod_figure_spec_register(unsigned (*get_spec_tag1)(),
                                       unsigned (*get_spec_tag2)(),
                                       multimethod_figure_pointer spec) {
```

```
  // printf("start _print_figure_spec_register\n");
  // Указатель на функцию фиксируется в массиве указателей
  unsigned tag = get_figure_number() * get_spec_tag1() + get_spec_tag2();
  _multimethod_figure_array[tag] = spec;
}
```

Добавление любого из обработчиков специализаций также осуществляется по единой схеме, обеспечивающей независимость от ранее написанного кода. Для регистрации мультиметода используется соответствующий конструктор

```
// Мультиметод для двух прямоугольников
static void
multimethod_figure_rectangle_rectangle(figure* p_f1, figure* p_f2) {
    printf("rectangle and rectangle\n");
}
// Регистрация мультиметода для двух прямоугольников
void  __attribute__
((constructor(301))) register_multimethod_figure_rectangle_rectangle() {
  _multimethod_figure_spec_register(get_figure_rectangle_tag,
                                    get_figure_rectangle_tag,
                                    multimethod_figure_rectangle_rectangle);
}
```

Дальнейшее расширение мультиметода осуществляется по аналогичной схеме.

Полный пример, демонстрирующий предлагаемых подход к реализации процедурно-параметрического полиморфизма с подробным описанием эволюционного расширения создаваемой программы представлен по ссылке [19].

4. Инструментальная поддержка процедурно-параметрической парадигмы

Приведенное выше моделирование эволюционного расширения программы на языке программирования C показывает, что основные подходы к реализации предлагаемых конструкций мало чем отличаются от тех, которые ранее были использованы при расширении языка Оберон-2, анализе методов реализации обобщенных записей, моделировании процедурно-параметрического полиморфизма на языке C++ [15—17, 20]. Вместе с тем, наличие для языка программирования C компиляторов и инструментальных средств, поддерживающих возможность интеграции с ними на уровне исходных текстов позволяет выбирать соответствующие инструменты и использовать их для повышения эффективности собственных разработок.

Одной из таких систем является семейство компиляторов clang [21]. Доступ к исходным текстам компилятора и предлагаемые программные интерфейсы позволяют непосредственно модифицировать синтаксический анализатор, обеспечивая добавление в него правил, распознающих синтаксис вводимых конструкций. На этапе семантического анализа имеется доступ к абстрактному синтаксическому дереву (АСД), структуру которого можно подкорректировать с учетом тех изменений, которые вносятся в семантику языка C процедурно-параметрическими расширениями. Эти модификации поддерживаются методами классов, используемых при формировании узлов АСД. При семантическом анализе предлагаемых конструкций в АСД добавляются объекты, расширяющие семантику обычных структур и функций до их процедурно-параметрических реализаций. При этом добавляемые объекты должны быть эквивалентны представленным выше конструкциям, описанным на языке C. Использование для изменения дерева классов, реализованных в `clang`, позволяет не изменять генератор кода из C в llvm [22], тем самым упрощая исходную задачу его формирования для дополнительно вводимых конструкций.

Заключение

Предлагаемое расширение языка программирования C конструкциями, обеспечивающими инструментальную поддержку процедурно-параметрической парадигмы программирования, позволяет разрабатывать эволюционно расширяемые программы. При этом обеспечивается добавление новых альтернативных данных, а также функций, позволяющих безболезненно для ранее написанного кода использовать множественный полиморфизм. Проведенное моделирование вносимых изменений, реализованное на языке программирования C в рамках операционной системы Linux подтверждает возможности такой реализации. При необходимости рассматриваемый подход может непосредственно использоваться для создания эволюционно расширяемых программ на процедурном языке программирования.

Реализация описанного решения возможна с использованием уже существующих инструментальных средств, таких как семейства компиляторов clang [21] и библиотеки libtooling [23], что позволяет осуществлять модификации как синтаксического анализатора, так и изменять в ходе анализа абстрактное синтаксическое дерево, адаптируя его под новые конструкции.

References

[1] D. Shafer and D. A. Ritz, *Practical Smalltalk. Using Smalltalk/V*, Springer-Verlag. 1991, p. 233.

[2] H. John, *Advanced Guide to Python 3 Programming*, Springer. 2019, p. 497.

[3] M. Gregoire, *Professional C++*, John Wiley & Sons. 2018, p. 1122.

[4] E. Sciore, *Java Program Design*, Apress Media. 2019, p. 1122.

[5] J. Albahari and B. Albahari, *C# 6.0 Pocket Reference: Instant Help for C# 6.0 Programmers*, O'Reilly Media. 2016, p. 224.

[6] A. Freeman, *Pro Go: The Complete Guide to Programming Reliable and Efficient Software Using Golang*, Apress. 2022, p. 1105.

[7] J. Blandy, J. Orendorff, and L. F. Tindall, *Programming Rust*, O'Reilly Media. 2021, p. 735.

[8] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns. Elements of Reusable Object-Oriented Software*, Addison-Wesley Professional. 1994, p. 416.

[9] A. Alexandrescu, *Modern C++ Design. Generic Programming and Design Patterns Applied.* Addison-Wesley Professional. 2001, p. 360.

[10] S. Meyers, *More effective C++. 35 New Ways to Improve Your Programs and Designs*, Addison-Wesley Professional. 1996, p. 318.

[11] A. Legalov, «OOP, Multimethods and Pyramidal Evolution», *Open Systems*, no. 3, pp. 41–45, 2002, In Russian.

[12] L. Demichiel, «Overview: The common lisp object system», *Lisp and Symbolic Computation*, no. 1, pp. 227–244, 1989.

[13] B. Stroustrup, *Design and Evolution of C++*, Addison-Wesley Professional. 1994, p. 480.

[14] A. Legalov, *Procedurally-parametric programming paradigm. Is it possible as alternative to the object-oriented style?*, Dep. hands. Number 622-V00 Dep. VINITI 13.03.2000. 2000, p. 43, In Russian.

[15] I. Legalov, «Using of generalized records in procedural-parametric programming language», *Scientific Bulletin of the NSTU*, vol. 28, no. 3, pp. 25–37, 2007, In Russian.

[16] A. Legalov and D. Schvetc, «Procedural language with support for evolutionary design», *Scientific Bulletin of the NSTU*, vol. 15, no. 2, pp. 25–38, 2003, In Russian.

[17] A. Legalov and P. Kosov, «Evolutionary extension of programs using the procedural-parametric approach», *Computational Technologies*, vol. 21, no. 3, pp. 56–69, 2016, In Russian.

[18] *Linux x86 Program Start Up or - How the heck do we get to main()?* [Online]. Available: http://dbp-consulting.com/tutorials/debugging/linuxProgramStartup.html.

[19] *An example of an evolutionary extension of a program using a procedural-parametric approach.* [Online]. Available: https://github.com/kreofil/c-evolution-example.

[20] A. Legalov, «Multimethods and paradigms», *Open Systems*, no. 5, pp. 33–37, 2002, In Russian.

[21] *Clang: a C language family frontend for LLVM.* [Online]. Available: https://clang.llvm.org/.

[22] *The LLVM Compiler Infrastructure.* [Online]. Available: https://llvm.org/.

[23] *LibTooling.* [Online]. Available: https://clang.llvm.org/docs/LibTooling.html.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 1, 2023
journal homepage: www.mais-journal.ru

THEORY OF DATA

Name Entity Recognition Tasks: Technologies and Tools

N. S. Lagutina[1], A. M. Vasilyev[1], D. D. Zafievsky[1]

DOI: 10.18255/1818-1015-2023-1-64-85

[1]P. G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 68T50
Research article
Full text in Russian

Received October 21, 2022
After revision February 1, 2023
Accepted February 8, 2023

The task of named entity recognition (NER) is to identify and classify words and phrases denoting named entities, such as people, organizations, geographical names, dates, events, terms from subject areas. While searching for the best solution, researchers conduct a wide range of experiments with different technologies and input data. Comparison of the results of these experiments shows a significant discrepancy in the quality of NER and poses the problem of determining the conditions and limitations for the application of the used technologies, as well as finding new solutions. An important part in answering these questions is the systematization and analysis of current research and the publication of relevant reviews. In the field of named entity recognition, the authors of analytical articles primarily consider mathematical methods of identification and classification and do not pay attention to the specifics of the problem itself. In this survey, the field of named entity recognition is considered from the point of view of individual task categories. The authors identified five categories: the classical task of NER, NER subtasks, NER in social media, NER in domain, NER in natural language processing (NLP) tasks. For each category the authors discuss the quality of the solution, features of the methods, problems, and limitations. Information about current scientific works of each category is given in the form of a table for clarity. The review allows us to draw a number of conclusions. Deep learning methods are leading among state-of-the-art technologies. The main problems are the lack of datasets in open access, high requirements for computing resources, the lack of error analysis. A promising area of research in NER is the development of methods based on unsupervised techniques or rule-base learning. Intensively developing language models in existing NLP tools can serve as a possible basis for text preprocessing for NER methods. The article ends with a description and results of experiments with NER tools for Russian-language texts.

Keywords: natural language processing; text features; automated essay scoring; business letter

INFORMATION ABOUT THE AUTHORS

Nadezhda Stanislavona Lagutina correspondence author	orcid.org/0000-0002-6137-8643. E-mail: lagutinans@gmail.com PhD, associate professor.
Andrey Mikhaylovich Vasilyev	orcid.org/0000-0001-6672-0981. E-mail: andrey@crafted.su associate professor.
Daniil Dmitrievich Zafievsky	orcid.org/0000-0001-8266-2283. E-mail: zafievsky@mail.ru student.

Funding: This work was supported by initiative program VIP-016.

For citation: N. S. Lagutina, A. M. Vasilyev, and D. D. Zafievsky, "Name Entity Recognition Tasks: Technologies and Tools", *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 64-85, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 1, 2023
сайт журнала: www.mais-journal.ru

THEORY OF DATA

Задачи в области распознавания именованных сущностей: технологии и инструменты

Н. С. Лагутина[1], А. М. Васильев[1], Д. Д. Зафиевский[1]

DOI: 10.18255/1818-1015-2023-1-64-85

[1]Ярославский государственный университет им. П. Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 004.912
Научная статья
Полный текст на русском языке

Получена 21 октября 2022 г.
После доработки 1 февраля 2023 г.
Принята к публикации 8 февраля 2023 г.

Задача распознавания именованных сущностей (named entity recognition, NER) состоит в выделении и классификации слов и словосочетаний, обозначающих именованные объекты, таких как люди, организации, географические названия, даты, события, обозначения терминов предметных областей. В поисках лучшего решения исследователи проводят широкий спектр экспериментов с разными технологиями и исходными данными. Сравнение результатов этих экспериментов показывает значительное расхождение качества NER и ставит проблему определения условий и границ применения используемых технологий, а также поиска новых путей решения. Важным звеном в ответах на эти вопросы является систематизация и анализ актуальных исследований и публикация соответствующих обзоров. В области распознавания именованных сущностей авторы аналитических статей в первую очередь рассматривают математические методы выделения и классификации и не уделяют внимание специфике самой задачи. В предлагаемом обзоре область распознавания именованных сущностей рассмотрена с точки зрения отдельных категорий задач. Авторы выделили пять категорий: классическая задача NER, подзадачи NER, NER в социальных сетях, NER в предметных областях, NER в задачах обработки естественного языка (natural language processing, NLP). Для каждой категории обсуждается качество решения, особенности методов, проблемы и ограничения. Информация об актуальных научных работах каждой категории для наглядности приводится в виде таблицы, содержащей информацию об исследованиях: ссылку на работу, язык использованного корпуса текстов и его название, базовый метод решения задачи, оценку качества решения в виде стандартной статистической характеристики F-меры, которая является средним гармоническим между точностью и полнотой решения. Обзор позволяет сделать ряд выводов. В качестве базовых технологий лидируют методы глубокого обучения. Основными проблемами являются дефицит эталонных наборов данных, высокие требования к вычислительным ресурсам, отсутствие анализа ошибок. Перспективным направлением исследований в области NER является развитие методов на основе обучения без учителя или на основе правил. Возможной базой предобработки текста для таких методов могут служить интенсивно развивающиеся модели языков в существующих инструментах NLP. Завершают статью описание и результаты экспериментов с инструментами NER для русскоязычных текстов.

Ключевые слова: распознавание именованных сущностей; автоматическая обработка текста; обзор; инструменты обработки естественного языка

ИНФОРМАЦИЯ ОБ АВТОРАХ

Надежда Станиславовна Лагутина
автор для корреспонденции
orcid.org/0000-0002-6137-8643. E-mail: lagutinans@gmail.com
канд. физ.-мат. наук, доцент.

Андрей Михайлович Васильев
orcid.org/0000-0001-6672-0981. E-mail: andrey@crafted.su
доцент.

Даниил Дмитриевич Зафиевский
orcid.org/0000-0001-8266-2283. E-mail: zafievsky@mail.ru
студент.

Финансирование: Работа выполнена при поддержке инициативного проекта VIP-016.

Для цитирования: N. S. Lagutina, A. M. Vasilyev, and D. D. Zafievsky, “Name Entity Recognition Tasks: Technologies and Tools”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 64-85, 2023.

Введение

Именованная сущность (named entity, NE) является идентификатором имен людей, названий организаций, географических местоположений, событий, дат и т. п. в тексте на естественном языке. Впервые термин был использован в 1996 году [1]. С этого момента NER стало классической задачей NLP.

Большое количество исследований, посвященных NER, показывает не только актуальность этой задачи, но и необходимость обобщения и анализа накапливаемых результатов. Практически все обзоры, как ранние [2, 3], так и поздние [4, 5] систематизируют методы выделения NE. При этом методы глубокого обучения получают наибольшее внимание, так как часто дают самые лучшие результаты. Однако соревнования между исследователями в области NLP демонстрируют, что для разных задач одни и те же методы NER показывают различное качество работы [6, 7]. Поэтому авторы данного обзора решили систематизировать результаты NER, отталкиваясь от категорий задач в этой области. Анализ отдельных задач позволит более детально понять границы применения методов выделения NE, определить перспективные направления исследований, увидеть спектр областей применения инструментов NER. Обзор содержит работы за период с 2018 по 2022 годы, но включает и некоторые более ранние, показавшиеся авторам этой статьи важными и интересными.

Каждая категория задач описана в отдельном разделе статьи и снабжена таблицей, агрегирующей информацию об исследованиях: ссылку на работу, язык использованного корпуса текстов, базовый метод решения задачи, название корпуса текстов, качество решения F-мера. Второй раздел посвящен классической задаче NER, таблица 1. Третий раздел рассматривает подзадачи NER, таблица 2. Четвертый — NER в социальных сетях, таблица 3. Пятый — NER в предметных областях, таблица 5. Шестой — NER в задачах NLP, таблица 6. В седьмом разделе описаны эксперименты с тремя инструментами NLP: spaCy, Stanford's NER (Stanza) и DeepPavlov, предоставляющими NER API для русского языка. Авторы не включили в обзор большое количество работ, посвященных китайскому языку, из-за лингвистических особенностей, отличающих его от других языков.

1. Классическая задача NER

Классическая задача NER ставит цель распознавать в тексте на естественном языке такие единицы как имена людей, названия организаций, географические местоположения, числовые выражения, включая время, дату, деньги, и классифицировать их по соответствующим категориям. С момента начала исследований три категории NE получили наибольшее распространение: имена людей (PER), географические местоположения (LOC), названия организаций (ORG), иногда их называют generic NE [4]. NE, которые не соответствуют ни одному типу, иногда выделяют в отдельную категорию miscellanea (MISC). В таблице 1 приведены значения F-меры для разных категорий сущностей и среднее значение — overall. Кроме трех указанных, в таблицу 1 вошли продукты (PROD) и события (EVENT).

Показательный результат решения задачи NER для английского языка можно увидеть в работе [8]. Для построения модели текста авторы применили рекуррентные нейронные сети (RNN). Для выбора архитектуры сети использовался подход на основе дифференцированного поиска (DARTS). Эксперименты проводились с англоязычным корпусом текстов CoNLL-2003, который содержит четыре типа NE: PER, LOC, ORG, MISC. Качество NER F-мера составила 93.47 %.

Близкий по качеству результат NER F-мера 90.79 % получен для английского языка и в более ранней работе [9]. Авторы интегрировали в ансамбль четыре инструмента: Stanford Named Entity Recognizer, Illinois Named Entity Tagger, Ottawa Baseline Information Extraction и Apache OpenNLP Name Finder. Работа показала, что обучение в ансамбле может снизить частоту ошибок. Авторы указали значение F-меры для всех трех категорий NE (см. табл. 1). Однако столь высокий результат может быть обусловлен качеством подготовленного собственного корпуса текстов.

Table 1. Research devoted to the classical task of NER

Таблица 1. Исследования, посвященные классической задаче NER

Авторы	Язык	Метод	Корпус текстов	F-мера, %					
				Overall	PER.	LOC.	ORG.	PROD.	EVENT
[8]	английский	RNN+DARTS	CoNLL-2003	93.47	-	-	-	-	-
[9]	английский	ensemble learning	авторский	90.79	95.91	91.27	85.18	-	-
[10]	английский	BiLSTM-CRF	OntoNotes5	87.95	-	-	-	-	-
	английский	BiLSTM-CRF	CoNLL-2003	91.73	-	-	-	-	-
[11]	английский	BiLSTM-CRF	CoNLL-2003	92.42	-	-	-	-	-
	немецкий	BiLSTM-CRF	CoNLL-2003	86.21	-	-	-	-	-
	голландский	BiLSTM-CRF	CoNLL-2003	88.25	-	-	-	-	-
[12]	немецкий	BiLSTM-CRF	CoNLL-2003	82.99	-	-	-	-	-
	немецкий	BiLSTM-CRF	GermEval	81.83	-	-	-	-	-
[13]	бразильский	LSTM-CRF	LeNER-Br	92.53	93.47	60.54	88.38	-	-
[14]	испанский	BiLSTM-CRF	CoNLL-2002	87.08	-	-	-	-	-
[15]	испанский	(NLTK+GATE)-LSTM	CoNLL-2002	92.0	-	-	-	-	-
[16]	русский	BiLSTM-CRF	Gareev's dataset	87.17	95.08	-	84.41	-	-
	русский	BiLSTM-CRF	Persons-1000	99.26	99.26	-	-	-	-
	русский	BiLSTM-CRF	FactRuEval 2016	82.10	-	-	-	-	-
[17]	датский	Danish BERT	DaNE	83.76	93.52	87.30	78.27	-	-
[18]	чешский	BERT-CRF	авторский	93.9	-	-	-	-	-
	русский	BERT-CRF	авторский	87.3	-	-	-	-	-
	болгарский	BERT-CRF	авторский	87.2	-	-	-	-	-
	польский	BERT-CRF	авторский	93.2	-	-	-	-	-
[19]	болгарский	BERT-CRF	авторский	89.6	93.9	94.8	85.1	59.5	50.0
	чешский	BERT-BiLSTM-CRF	авторский	94.4	95.7	98.3	95.0	79.6	55.0
	польский	BERT-CRF	авторский	93.7	93.0	95.5	95.5	54.1	100.0
	русский	rule-based+ ML-techniques	авторский	86.1	93.3	98.7	92.5	65.1	-
[20]	арабский	BERT	AQMAR	65.5	74.4	68.4	34.6	-	-
	арабский	BERT	NEWS	78.6	89.5	80.5	60.8	-	-
[21]	английский	BERT+self-training	CoNLL-2003	81.48	-	-	-	-	-
	английский	BERT	Tweet	48.01	-	-	-	-	-
	английский	BERT	OntoNotes5	68.35	-	-	-	-	-
	английский	BERT	Webpage	65.74	-	-	-	-	-
	английский	BERT	Wikigold	60.07	-	-	-	-	-
[22]	португальский	Local Grammar	PrimeiroHarem	59.0	-	-	-	-	-
[23]	турецкий	rule-based	авторский	88.71	88.58	84.71	91.47	-	-
[24]	финский	rule-based	Digitoday	86.82	85.32	93.51	89.35	76.41	100.00
	финский	BiLSTM-CRF	Digitoday	84.59	80.90	89.76	88.62	74.23	80.00
[25]	малайский	ELMO+MTBR	Malay NER	96.32	-	-	-	-	-

Более надежно выглядит метод, показывающий стабильно высокие результаты на разных корпусах [10]. Качество результата показало F-меру 87.95 % для набора данных OntoNotes с 18 типами сущностей и 91.73 % для набора данных CoNLL-2003. Архитектура классификатора представляет собой популярный Bi-LSTM-CRF. Основой вектора числовых характеристик текста являются эмбеддинги слов, а также параметры символьного уровня.

Такая же архитектура нейронных сетей Bi-LSTM-CRF лежит в основе исследований NER для трех европейских языков [11]. Лучший результат оказался для текстов на английском: F-мера 92.42 %, немного ниже на голландском: F-мера 88.25 % и на немецком F-мера 86.21 %. Аналогичный подход использовался для немецкого языка в работе [12]. F-мера оказалась ниже — около 82 %, но почти одинакова для двух разных наборов данных.

Следует отметить очевидную популярность нейронных сетей, моделирующих долгую кратковременную память (LSTM), в частности двунаправленную (Bi-LSTM), в решении задачи NER. Их комбинация с условными случайными полями, вероятностными моделями для сегментации и мар-

кировки последовательностей данных (CRF), стала стандартом для получения глубоких контекстно-зависимых представлений текста [4]. Эта технология показывает высокий результат NER для бразильских [13] F-мера 92.53 % и испанских [14] F-мера 87.08 % текстов. Интересно, что для испанского языка более высокий результат (F-мера 92.0 %) получился после предварительного отбора NE с помощью стандартных инструментов для обработки текста: NLTK и GATE [15]. Для русского языка эта технология также дает хорошее качество решения задачи [16].

Нейронные сети лежат в основе нейросетевой модели естественных языков BERT. В последнее время BERT широко используется в задачах NLP, в том числе NER. Эта модель показала хорошее качество NER для датского языка F-мера 83.76 % [17]. Комбинация технологий BERT-CRF оказалась эффективна для четырех славянских языков [18, 19]. Детальные результаты этих исследований размещены в таблице 1.

Однако применение нейронных сетей невозможно без предварительно размеченных корпусов данных большого объема. Этот факт накладывает существенные ограничения на использование указанной технологии в случае малого количества текстов, в следствие дорогостоящего сбора и разметки. Поэтому менее требовательные к ресурсам методы остаются привлекательными для исследователей.

Исследователи [21] изучали проблему распознавания именованных объектов (NER) с помощью технологии опосредованного обучения. Опосредованное обучение не требует большого количества ручных аннотаций и позволяет получать маркеры текста через внешние базы знаний, однако в ряде случаев маркеры слов оказываются ошибочными или отсутствуют. Авторы предлагают новую вычислительную структуру — BOND, использующую модели BERT и RoBERTa. BOND использует предварительно обученные языковые модели общего назначения, что позволяет ей работать в условиях ограниченных вычислительных ресурсов и обучающих корпусов. Однако хорошее качество решения оказалось только для одного корпуса CoNLL-2003 F-мера 81.48 %.

Оливера и др. [22] разработали локальную грамматику для NER на португальском языке (Local Grammar). Однако качество решения задачи оказалось низким, F-мера 59.0 %. Гораздо лучше показали себя методы, основанные на правилах (rule-based) для турецких текстов, F-мера 88.71 % [23], русских, F-мера 86.1 % [19] и финских F-мера 86.82 % [24]. Для финского языка метод, основанный на правилах, даже превзошел популярную технологию BiLSTM-CRF, F-мера 84.59 %.

Таким образом, задача NER достаточно хорошо решена для многих естественных языков. Однако это можно сказать уверенно только для небольшого количества категорий сущностей, в первую очередь для PER, LOC, ORG. Другие категории, такие как PROD, EVENT, DATE исследованы намного меньше даже для английского языка. Частично это связано с отсутствием размеченных корпусов или их малым объемом.

Исследователи в поисках лучшего решения часто проводят широкий спектр экспериментов с разными корпусами и разными технологиями. Сравнение результатов этих экспериментов показывает расхождение в качестве NER. Этот факт ставит проблему определения условий и границ применения используемых технологий, а также поиска новых решений. Один из путей преодоления таких проблем: анализ ошибок. К сожалению, понять и объяснить процессы происходящие в нейронных сетях трудно, более перспективными для этой цели кажутся методы основанные на правилах.

2. Подзадачи NER

Качественное выделение стандартных типов NE может внести существенный вклад в анализ естественного языка и стать основой решения задач в области информационного поиска. Повышение эффективности решения сложной задачи невозможно без подробного анализа ее деталей. Поэтому в этом разделе мы обсудим подзадачи, возникающие в сфере NER.

Table 2. NER subtasks research

Таблица 2. Исследования, посвященные подзадачам NER

Авторы	Задача	Язык	Метод	Корпус текстов	F-мера, %
[26]	вложенные NER	английский	BiLSTM-CRF	GENIA	74.79
	вложенные NER	английский	BiLSTM-CRF	ACE 2005	72.25
[27]	вложенные NER	английский	BiLSTM-Detection Network-Classifier	ACE 2005	78.2
[28]	вложенные NER	английский	Span-level Graph	ACE 2005	85.11
[29]	вложенные NER	английский	CNN-BERT-BiLSTM	OntoNotes5	91.3
	вложенные NER	английский	CNN-BERT-BiLSTM	CoNLL-2003	93.5
	вложенные NER	немецкий	CNN-BERT-BiLSTM	CoNLL-2003	86.4
	вложенные NER	испанский	CNN-BERT-BiLSTM	CoNLL-2002	90.3
	вложенные NER	датский	CNN-BERT-BiLSTM	CoNLL-2002	93.7
[30]	вложенные NER	английский	BiLSTM	GENIA	77.1
	вложенные NER	английский	BiLSTM	JNLPBA	78.4
[31]	границы NE	английский	CNN-BiLSTM-GRU	OntoNotes5	93.94
[32]	границы NE	иврит	BiLSTM-CRF	HAARETZ	85.22
[33]	нормализация NE	английский	BERT+NN	NCBI, BC5CDR	89.1
[34]	нормализация NE	английский	BioBERT	NCBI, BC5CDR	86.6

NER в условиях ограниченных наборов обучающих данных можно рассматривать как отдельную важную подзадачу выделения NE. В работе [35] эта проблема решается с помощью адаптации прототипической нейронной сети (prototypical network), специальной модели, разработанной для классификации в условиях, когда размеченных примеров мало. Эта сеть отображает объекты в векторное пространство, в котором классификация может быть выполнена путем вычисления расстояний до прототипных представлений каждого класса. В экспериментах авторы строят корпус текстов на основе данных из Ontonotes с 18 размеченными категориями NE и уменьшают количество примеров обучающей выборки до 20 для отдельной категории. Качество распознавания при этом сильно отличается для каждого типа NE. Один из лучших результатов получается для PER: F-мера 82.32 %, ORG: 69.2 %, LOC: 52.0 %, EVENT: 45.2 %.

Среди других подзадач самое большое внимание уделяется выделению вложенных именованных сущностей (nested named entity recognition). Например, фразы «Банк Китая» и «Университет Вашингтона» являются вложенными именованными объектами, где в название организации вложено местоположение [36]. Учёные [26] предложили новую нейронную модель для идентификации вложенных объектов путем динамического наложения слоев flatNER. Каждый слой включает LSTM и CRF, которые создают новое представление для обнаруженных объектов, а затем передают их в следующий слой. Модель динамически складывает слои flatNER до тех пор, пока объекты извлекаются. Оценка показывает, что предлагаемый подход превосходит системы, основанные на стандартных методах NER, достигая F-меры 74.7 % и 72.2 % в наборах данных GENIA и ACE2005 соответственно.

Авторы статьи [27] представляют инструмент MGNER для многозначного распознавания именованных сущностей. Он справляется со случаями, когда несколько сущностей в предложении могут быть перекрывающимися или полностью вложенными. Текст моделируют эмбеддинги слов, полученные на основе их морфологических характеристик (POS-тегов), информации о словах на уровне символов и встраивания языковой модели ELMo [37]. Качество этого решения немного выше, чем у предыдущей работы. Еще выше показывает результат алгоритм, основанный на графах [28].

В работах [29] и [30] для выявления вложенных NE определяют все возможные фразы, которые потенциально могут быть именованными сущностями, и классифицируют их с помощью глубоких нейронных сетей. В обоих случаях применяется сеть BiLSTM. Однако авторы первой работы [29] применяют этот подход для классической задачи NER для четырех европейских языков с использованием эмбеддингов на основе CNN и BERT, а второй [30] – работают со специфическими NE

медицинской области. Качество выделения NE оказывается выше в среднем на 13 %, это видно из таблицы 2.

Подход, использованный в двух предыдущих работах, связан с еще одной подзадачей NER: выделение границ NE. Исследователи [31] отмечают, что ошибки, допущенные при обнаружении границ объектов, отрицательно влияют на последующие системы выделения и классификации NE. Кроме того, они обращают внимание, что инструмент, качественно определяющий границы NE без явной детализации категории, может быть полезен в качестве предварительной обработки текста для последующего анализа в зависимости от предметной области. Авторы строят ансамбль нейронных сетей для детекции границ NE и получают высокое качество результата: F-мера 93.94 %.

Израильские учёные [32] также обращают внимание на проблему выделения границ NE, но в контексте морфологически богатых языков. Они предлагают показатель, который содержит аннотации NER на уровне лексем и морфем для текстов на иврите. Исследователи используют стандартные технологии NER BiLSTM-CRF, добавляя к характеристикам уровня символов и слов морфологические параметры. Свой подход авторы оценивают для корпуса еврейских исторических текстов и получают F-меру 85.22 %.

Еще одной подзадачей в области NER является нормализация именованных сущностей (NEN), т. е. приведение выделенной NE к некоторой начальной форме. Это необходимо для сопоставления и идентификации объектов в тех случаях, когда одна и та же NE имеет синонимы, полное название и аббревиатуру, разные названия. Авторы работы [33] предлагают автоматизированную модель нормализации именованных сущностей на основе новой нейронной сети, строящей и сопоставляющей графы отношений между NE. Сопоставление и обновление весов ребер позволяет получить информацию о семантическом сходстве между совпадающими сущностями. В качестве параметров сущностей используется модель BERT. Та же задача для медицинских текстов решается учеными в работе [34] с использованием BioBERT и математических классификаторов.

Одно из направлений развития методов NER лежит в области применения сложных морфологических, синтаксических и семантических параметров текста. Авторы работы [38] указывают, что в области NER мало исследованы лексические характеристики текстов. Они добавляют к параметрам текста из обычной модели LSTM-CRF характеристики сходства слов с типами сущностей. Исследователи показывают, что качество NER сравнимо со стандартными подходами, но производительность их системы выше. На необходимость использования синтаксической информации обращается внимание и в работе [39].

3. NER в социальных сетях

Table 3. Research about NER in social media

Таблица 3. Исследования, посвященные NER в социальных сетях

Авторы	Язык	Метод	Корпус текстов	F-мера, %				
				Overall	PER.	LOC.	ORG.	MISC
[40]	английский	CNN-BiLSTM-CRF	WNUT2017	41.86	58.66	52.86	26.55	-
[41]	английский	CNN-BiLSTM-CRF	TWITTER-2015	70.69	81.98	78.95	53.07	34.02
[42]	испанский	BiLSTM-CRF	ProfNER Gold Standard	83.9	-	-	-	-
[43]	английский	InceptionV3-BERT-LSTM-CRF	TWITTER-2015	73.47	86.44	77.16	52.91	36.05
[44]	английский	BERT-MMI-CRF	TWITTER-2015	73.41	85.24	81.58	63.03	39.45
	английский	BERT-MMI-CRF	TWITTER-2017	85.31	91.56	84.73	82.24	70.10
[20]	арабский	BERT	авторский	57.3	-	-	-	-
[45]	персидский	BERT	ParsNER-Social corpus	89.65	-	-	-	-
[46]	индийский	Transformer+CRF	авторский	49.79	-	-	-	-
[47]	английский	CRF + PLP	комментарии Yahoo!	71.0	-	-	-	-
[15]	английский	(NLTK+GATE)-LSTM	авторский	90.0	-	-	-	-

Среди всех работ в области NER как отдельную задачу можно выделить NER в социальных сетях. С одной стороны, ее постановка совпадает с классической, с другой стороны, особенности текстов в этой области приводят к отличиям в технологиях и качестве решений.

Использование самой распространенной комбинации BiLSTM-CRF для выделения четырех стандартных категорий сущностей: PER, LOC, ORG, MISC, не приводит к хорошим результатам. Это показывают меры качества из работ [40, 41], приведенные в таблице 3. Скорее всего, это объясняется качеством текстов социальных сетей и специфическим стилем их авторов. В частности на эту особенность указывают исследователи [40], описывая корпус WNUT2017. Однако для испанского языка технология BiLSTM-CRF показала результат гораздо лучше, чем для английского [42]: F-мера 83.9 % против 70.69 %. Возможно это также связано с особенностями корпуса текстов.

Языковая модель BERT и дополнительные характеристики, извлеченные из изображений, сопровождающих тексты, позволили повысить результаты для английского языка. Для анализа изображений использовались специальные инструменты: InceptionV3 [43] (F-мера 73.47 %) и MMI [44]. F-мера оказалась равна 73.41 %.

BERT стал основой NER в социальных сетях для арабских и иранских исследователей. Однако качество распознавания арабских текстов низкое: F-мера 57.3 % [20], а иранских вполне хорошее: F-мера 89.65 % [45].

Исследователи [46] обратили внимание, что индийские твиты включают много английских слов на латинице, в частности, при обозначении именованных сущностей. Для NER ученые применили новую для момента выполнения работы архитектуру нейронной сети Transformer [48] вместо обычной BiLSTM с целью повышения производительности. Однако качество решения задачи оказалось совсем низким F-мера 49.79 %. Скорее всего это связано с сильно специфическим стилем корпуса текстов.

Нестандартный подход к NER в комментариях пользователей описан в работе [47]. Предлагаемый инструмент основан на графе кореферентности меток NE и алгоритме распространения меток для выявления именованных сущностей. F-мера 71 % этого решения сравнима с показателями области в целом.

Самый хороший результат был получен турецкими учёными [15] с показателем F-меры 90.0 %. Их метод состоит из двух этапов. Во-первых, обработка текста инструментами GATE и NLTK, в результате которой обеспечиваются входные данные для второго этапа. Во-вторых, использование обученной рекуррентной нейронной сети для определения NE.

Одной из основных проблем NER в социальных сетях является зашумлённость исходного текста [40, 47]. Однако, на наш взгляд, не менее важной проблемой является постоянное возникновение новых сущностей [15], так как многие стандартные технологии базируются на лексических ресурсах: словарях, Википедии и т. п. Эти ресурсы обновляются медленно. Поэтому решением проблемы может быть дорогостоящее постоянное переобучение или создание методов не зависящих от обучающих корпусов, например, тщательно проработанных методов, основанных на правилах и использующих языковые модели, определяющие маркеры слов с высоким качеством.

4. NER в предметных областях

Распознавание именованных сущностей в предметных областях существенно отличается от классической задачи NER. В последнем случае хорошо работают методы, фиксирующие семантику слов на основе простых текстов и общих особенностей естественного языка. В случае NER в предметной области этого недостаточно [76]. Выявление самих NE и их категорий требует использования информации специфичной для конкретной области.

Самое большое количество исследований NER посвящено биомедицине. Это связано со значительным количеством доступных словарей и размеченных корпусов текстов: о заболеваниях NCBI-Disease, с названиями генов и белков BC2GM and JNLPBA, о химических веществах BC4CHEMD,

Table 4. Biomedical NER research

Таблица 4. Исследования, посвященные NER в биомедицине

Авторы	Язык	Метод	Корпус текстов	F-мера, %
[49]	английский	BiLSTM-CRF	BC2GM	80.74
	английский	BiLSTM-CRF	BC4CHEMD	89.37
	английский	BiLSTM-CRF	JNLPBA	73.52
	английский	BiLSTM-CRF	BC5CDR	88.78
	английский	BiLSTM-CRF	NCBI-Disease	86.14
[50]	английский	BiLSTM-CRF	BC2GM	79.73
	английский	BiLSTM-CRF	BC4CHEMD	88.85
	английский	BiLSTM-CRF	JNLPBA	77.58
	английский	BiLSTM-CRF	BC5CDR-chem	93.31
	английский	BiLSTM-CRF	BC5CDR-disease	84.08
	английский	BiLSTM-CRF	NCBI-Disease	86.36
[51]	английский	BiLSTM-CRF	CRAFT	72.31
	английский	BiLSTM-CRF	BioNLP CG	78.20
	английский	BiLSTM-CRF	PDR	83.44
	английский	BiLSTM-CRF	JNLPBA	77.78
	английский	BiLSTM-CRF	BC5CDR	90.57
	английский	BiLSTM-CRF	NCBI-Disease	87.47
[52]	английский	BERT+rule-based	BC2GN	86.7
	английский	BiLSTM-CRF	BC3GN	50.1
	английский	BiLSTM-CRF	OSIRISv1.2	88.14
	английский	BiLSTM-CRF	Thomas corpus	89.08
[53]	английский	Multi-BERT	EN EHR	92.39
	английский	Multi-BERT	EN UGT	84.88
	русский	Multi-BERT	RU UGT	60.45
	русский	Multi-BERT	RU EHR	85.00
[54]	итальянский	BiLSTM-CRF	SIRM COVID-19	89.01
	итальянский	BERT	SIRM COVID-19	88.58

BC5CDR. Первые три исследования, приведенные в таблице 4, используют классическую технологию NER BiLSTM-CRF для указанных корпусов текстов. Качество распознавания NE стабильно высокое: F-мера 73 % - 93 %. Интересно, что наблюдается корреляция между F-мерой в разных работах и соответствующими корпусами. Это показывает, что технология NER, базирующаяся на глубоком обучении, вполне подходит для предметной области и зависит от качества и объема обучающего корпуса текстов.

Другая технология, сочетающая языковую модель BERT и правила, также показывает хорошие результаты: F-мера 86 % - 89 % [52]. Как основа решения задачи BERT использован и в работе [53]. Исследователи определяли названия заболеваний и медикаментов для текстов на английском и русском языках. Лучшие результаты получились для английского: F-мера 92.39 %. Авторы сравнили технологии BERT и BiLSTM-CRF и показали преимущество первого, в частности для русского языка. Однако технология BiLSTM-CRF была успешно применена для итальянского [54]. В работе [77] также выявлено преимущество модели BERT. Однако реальное применение методов в промышленных программах требует оценки эффективности с точки зрения времени работы и используемой памяти [64].

Хорошие результаты NER дает классический подход BiLSTM-CRF в химической области [55, 56, 78], см. таблицу 5. А также в извлечении NE из научных статей [57] и в материаловедении [58, 60].

Интересно, что в текстах о еде лучшие показатели F-меры, более 95 %, оказались у методов, основанных на правилах [62, 65]. Возможно это связано с особенностями структуры текстов, используемых в основе корпусов, например, рецептов [63]. Хотя и стандартный подход показывает высокое качество F-меры: 92.5 % [61]. Методы, основанные на правилах, были использованы

Table 5. Domain NER research

Таблица 5. Исследования, посвященные NER в предметных областях

Авторы	Область	Язык	Метод	Корпус текстов	F-мера, %
[55]	химия	английский	Att-BiLSTM-CRF	CHEMDNER	90.84
[56]	химия	английский	BiLSTM-CRF	CHEMDNER	90.0
[57]	наука	английский	BiLSTM-CRF	Macromolecules+ICPSR	90.9
[58]	материаловедение	английский	BiLSTM-CRF	авторский	87.04
[59]	производство	английский	BiLSTM-CRF	авторский	88.0
[60]	материаловедение	английский	BiLSTM-CRF	авторский	91.66
[61]	еда	английский	BERT-BiLSTM-CRF	YELP	92.5
[62]	еда	английский	rule-based	Allrecipes+MyRecipes	96.05
[63]	еда	английский	BERT+BiLSTM+Dict	авторский	81.31
[64]	продукты	английский	Dict+CRF	FoodBase	97.4
[65]	диетология	английский	rule-based	Documents from USDA	97.0
[66]	садоводство	английский	rule-based+CRF	article abstracts of journals	81.04
[67]	кибербезопасность	английский	rule-based+BiLSTM-CRF	авторский	75.05
[68]	кибербезопасность	английский	BERT-BiLSTM-CRF	Bridges dataset	96.87
[69]	кибербезопасность	русский	BERT	авторский	72.74
[70]	юриспруденция	английский	BERT-CRF	EDGAR-NER	96.1
[71]	юриспруденция	турецкий	BiLSTM	авторский	92.27
[72]	археология	голландский	BERT-CRF	авторский	73.5
[73]	география	французский	(CasEN+CoreNLP)-CRF	GeoNER-corpus	85.2
[74]	финансы	английский	dependency parse trees+reg_exp	TU Delft's repository	80.0
[75]	биомедицина	английский	CNN-BiGRU-CRF	BioNLP13PC	85.11
	кино	английский	CNN-BiGRU-CRF	MIT Movie	86.41
	рестораны	английский	CNN-BiGRU-CRF	MIT Restaurant	78.05
	безопасность	английский	CNN-BiGRU-CRF	Re3d	68.52
	финансы	английский	CNN-BiGRU-CRF	SEC	83.44
[76]	биомедицина	английский	entity selector-CRF	NCBI	86.12
	общие	английский	entity selector-CRF	CoNLL03	92.40
	биология	английский	entity selector-CRF	GENIA	75.59
	финансы	английский	entity selector-CRF	SEC	75.59

также для отбора кандидатов на роль NE для анализа аннотаций статей в области садоводства на английском: F-мера 81.04 % [66].

Авторы работы [70] выделяли NE в юридических текстах. Они использовали модель BERT обученную на общем англоязычном корпусе CoNLL-2003 и на специализированном EDGAR-NER и сравнивали результаты с более ранними моделями. Ранние модели показали невысокую эффективность в обоих случаях. Для более продвинутых моделей CRF и BERT, обученных на юридических текстах, качество распознавания существенно превышает случай обучения тех же моделей на общих английских текстах. Однако авторы не могут сказать, связано ли с различиями в моделях или различиями в тестовых коллекциях.

Сочетание методов, основанных на правилах, и технологий глубокого обучения было использовано для NER в области кибербезопасности [67]. Однако качество оказалось не слишком высоким: F-мера 75.05 %. Намного лучше результаты в этой области дало использование BERT для текстов на английском языке: F-мера 96.87 % [68]. Для русского языка учёные отметили, что аннотированный набор данных содержит небольшое количество объектов этих типов [69], и предложили автоматический метод увеличения набора данных с использованием BERT. В итоге качество NER оказалось 72.74 %.

Близкий по качеству результат получили исследователи археологических текстов на датском языке: F-мера 75.0 % [72]. В этой работе авторы применили BERT для получения характеристик слов и классификатор CRF для определения категории NE. Похожий подход был использован для выявления географических названий во французских текстах, но с более высоким результатом: F-мера

85.2 % [73]. Определение кандидатов NE осуществлялось с использованием существующих инструментов CasEN, CoreNLP, Perdido, SEM and spaCy. Дерево синтаксических зависимостей, построенное с помощью SpaCy для получения текстовых шаблонов, используется в работе [74]. Авторы объединили эту информацию с регулярными выражениями и разработали AckNER, инструмент для выделения финансовой информации.

Из анализа статей мы видим большое разнообразие предметных областей, выбранных для NER. Так же как и в классической задаче, методы на основе глубокого обучения дают хорошие результаты при наличии больших размеченных корпусов текстов, используемых для обучения. В большинстве предметных областей таких данных мало [75]. Интересно, что во многих исследованиях источником текстов являются научные статьи. Скорее всего это связано с эффективностью сбора и разметки данных. Для небольших корпусов лучше работают методы, основанные на правилах, но они сильно зависят от особенностей языка и структуры текста в целом.

Проблемы разнообразия предметных областей исследователи пытаются решить созданием универсальных инструментов, пригодных для использования в разных сферах жизни. В работе [75] предлагается систему, уже обученную для NER, адаптировать к новой предметной области. Другие исследователи [76] строят инструмент на основе базы знаний. Интересно, что в обоих случаях средняя F-мера похожи: 80.3 % и 84.82 % соответственно, но во втором всё-таки выше. Качество работы этих двух систем для разных предметных областей приведено в конце таблицы 5.

Кроме того, в последних по времени работах приобретают популярность уже существующие инструменты для NLP. Это связано с развитием качества их работы, оно существенно увеличилось буквально за последние несколько лет. Заметно так же, что подавляющее большинство работ выполнено для текстов на английском языке. Это дает богатый спектр исследовательских задач для национальных языков.

5. NER в задачах NLP

Table 6. Research devoted to NER in NLP tasks

Таблица 6. Исследования, посвященные NER в задачах NLP

Authors	Task	Language	Method	Text corpora	F-мера, %
[79]	аннотирование	английский	spaCy	English-Inshorts	-
[60]	аннотирование	английский	BiLSTM	авторский	91.4
[80]	аннотирование	английский	BiLSTM-CRF	авторский	89.35
[81]	аннотирование	русский	-	авторский	-
[82]	тематическая классификация	английский	weighting scheme NE	авторский	-
[83]	тематическая классификация	английский	BiLSTM + CRF	авторский	88.0
[84]	машинный перевод	английский немецкий	flairNLP	субтитры	-
[85]	анализ тональности	английский	Stanford Named Entity Tagger	SES, MSR2016	-
[86]	анализ тональности	английский	LSTM	авторский	-
[63]	онтология	английский китайский	BERT-BiLSTM-CRF	авторский	81.31
[87]	вопросно-ответные системы	английский	BERT	авторский	-
[88]	определение лжи	английский	spaCy, Stanford's NER	отзывы	-

С одной стороны, NER является классической самостоятельной задачей NLP. С другой стороны, есть ряд задач в области обработки текста и извлечения информации, в которые NER входит как подзадача или часть технологии решения. В этом разделе мы рассмотрим примеры таких задач. Для тех исследований, в которых NER рассматривается как выделенная подзадача, в таблице 6 приведена оценка качества ее решения. Остальные работы имеют другую цель и используют инструменты NER как часть технологии достижения этой цели и качество непосредственно NER не оценивают.

Сингх и др. [79] решают задачу аннотирования текста и отмечают, что одной из основных проблем классических алгоритмов является потеря именованных объектов. В работе предложен метод сохранения именованных объектов, которые полезны при создании точных аннотаций. В качестве инструмента NER исследователи используют spaCy. В другой работе [60] рассматривают NER как отдельную подзадачу аннотирования текста и посвящают ей свое исследование. Для решения используется нейронная сеть Bi-LSTM. Аналогичную задачу в коммерческих документах решают авторы [80]. Именованные сущности играют важную роль в системе аннотирования русскоязычных текстов [81].

Авторы работы [82] отмечают важную роль именованных объектов в задаче классификации текстов по тематике. В статье рассматривается байесовская версия алгоритма LDA. Исследователи модифицируют алгоритм, увеличивая веса NE и показывают, что именованные сущности хорошо подходят для использования в качестве терминов, специфичных для предметной области. Kumar A., Starly B. [83] отмечают, что одной из серьезных проблем при анализе текстов промышленного производства является отсутствие модели распознавания специфичных именованных объектов. Они предлагают собственную модель NER для классификации производственных текстов по категориям.

Участники конференции по переводу речи IWSLT 2021 [84] используют инструмент flairNLP для распознавания именованных сущностей, чтобы повысить качество машинного перевода субтитров. Другая специализированная система Stanford Named Entity Tagger, как часть решения задачи аспектного анализа тональности, применяется в работе [85]. Исследователи [86] для той же задачи строят собственную систему распознавания на основе LSTM.

Китайские учёные [63] использовали стандартную технологию, сочетающую BERT, BiLSTM и CRF для NER как часть системы построения графа предметной области. Разработанный AliMe KG, граф знаний в электронной коммерции, помогает понять потребности пользователей, ответить на их вопросы и создать поясняющие тексты, в том числе руководство по покупкам, ответы на вопросы о недвижимости, создание точек продаж. Другое исследование [87] напрямую изучает вопрос NER в вопросно-ответных системах.

Сложная цель построения системы автоматической детекции лжи была поставлена в работе [88]. Авторы используют именованные сущности для моделирования трех принципов: правдивые рассказчики предоставляют более подробные отчеты, правдивые тексты содержат больше упоминаний конкретных лиц, мест и дат, а обманщики склонны скрывать потенциально проверяемую информацию. Основой моделирования является доля именованных объектов, полученных с помощью двух инструментов spaCy и Stanford's NER.

Описанные работы показывают очень широкие возможности применения технологий NER. Особенно стоит обратить внимание на уже существующие инструменты NLP. Так же как и для извлечения именованных сущностей в предметных областях, они часто используются современными исследователями. Этот факт послужил мотивацией для сравнительного анализа качества работы spaCy, Stanford's NER (Stanza) и DeepPavlov для русского языка, родного для авторов обзора.

6. Эксперименты

6.1. Инструменты обработки естественного языка

Для анализа авторы выбрали инструменты, которые активно используются в научных исследованиях академическими кругами и в коммерческих приложениях. Актуальность и качество инструментов обеспечивается активностью их разработчиков, которая выражается в постоянном обновлении программной реализации.

spaCy — это библиотека Python с открытым исходным кодом, которая предоставляет инструменты для многих задач NLP, включая подсистему NER [89]. Библиотека достаточно гибкая и позволяет использовать для анализа текста как подходы основанные на правилах, так и на нейронных сетях.

Пользователь может либо использовать существующую текстовую модель, либо обучить новую. spaCy предоставляет три предварительно обученные модели для русского языка, которые нацелены на различные варианты баланса между требованиями к качеству и вычислительной мощности. В данном исследовании мы использовали самую большую доступную модель, `ru_core_news_lg`, которая должна обеспечивать наилучшее качество. Часть модели NER была обучена с использованием размеченного корпуса текстов Nerus[1] и реализована с использованием рекуррентной нейронной сети.

Stanford NLP Group предоставляет пакет Stanza NLP, который включает инструменты для анализа текста и предварительно обученные модели для более чем 70 языков [90]. Подсистема инструментария NER включает в себя модели для 17 языков. Модель распознавания именованных сущностей использует слои BiLSTM с представлениями на уровне символов и слов, а также слой декодирования CRF для получения прогнозов NER. Для русского языка Stanza NLP предоставляет единственную модель NER, которая была обучена на базе данных WikiNER [91]. Инструмент также предоставляет средства для обучения новой модели.

Последним инструментом, использованным в экспериментах, является диалоговый фреймворк DeepPavlov [92]. Фреймворк включает в себя теггер NER и набор предварительно обученных моделей. Модели могут использовать различные конструкции нейронных сетей, включая рекуррентные нейронные сети, BERT или гибридные. DeepPavlov предоставляет три предварительно подготовленные модели для русского языка. Все они были обучены с использованием набора данных Collection3 [93]. Для нашего исследования мы решили использовать модель `ner_rus_bert_torch`, поскольку авторы инструмента считают ее лучшей моделью, а также используют BERT, который не используется в других инструментах.

Использование описанных инструментов и моделей должно позволить проверить характеристики различных конструкций нейронных сетей в задаче NER для русского языка.

6.2. Корпус текстов

Чтобы оценить качество инструментов, мы не могли задействовать общедоступные наборы данных, поскольку они использовались для обучения моделей NER и дали бы значительное преимущество для соответствующего инструмента. Поэтому для использования в экспериментах мы сформировали собственный корпус текстов. Этот корпус включает тексты из различных источников: статьи из Википедии, новостные статьи, биографии исполнителей и тексты песен. Таким образом, эксперимент покажет качество NER для произвольного авторского текста из русского Интернета.

Корпус текстов, который мы собрали, содержит 50 текстов из различных источников. В каждом тексте мы вручную разметили именованные объекты для категорий PER, LOC, ORG и MISC. Количество слов и NE для каждого текста показано на рис. 1. В целом корпус содержит 616 LOC, 680 PER, 698 ORG и 966 объектов MISC.

Количество категорий было ограничено теми, которые предоставляют обученные модели. Модель Stanza может распознавать сущности во всех четырех категориях, но модели spaCy и DeepPavlov выделяют только категории PER, LOC и ORG.

6.3. Архитектура испытательного стенда

Оценка инструментов была полностью автоматизирована с использованием скриптов Python[2]. Эти скрипты применяют выбранные инструменты и анализируют результат распознавания именованных сущностей. Поток данных внутри испытательного стенда показан на рис. 2.

Каждый выбранный инструмент является библиотекой Python и, следовательно, должен быть встроен в приложение для выполнения. Для каждого из инструментов мы разработали

[1] https://github.com/natasha/nerus

[2] https://github.com/yarfruct/ner-russian-language-experiments

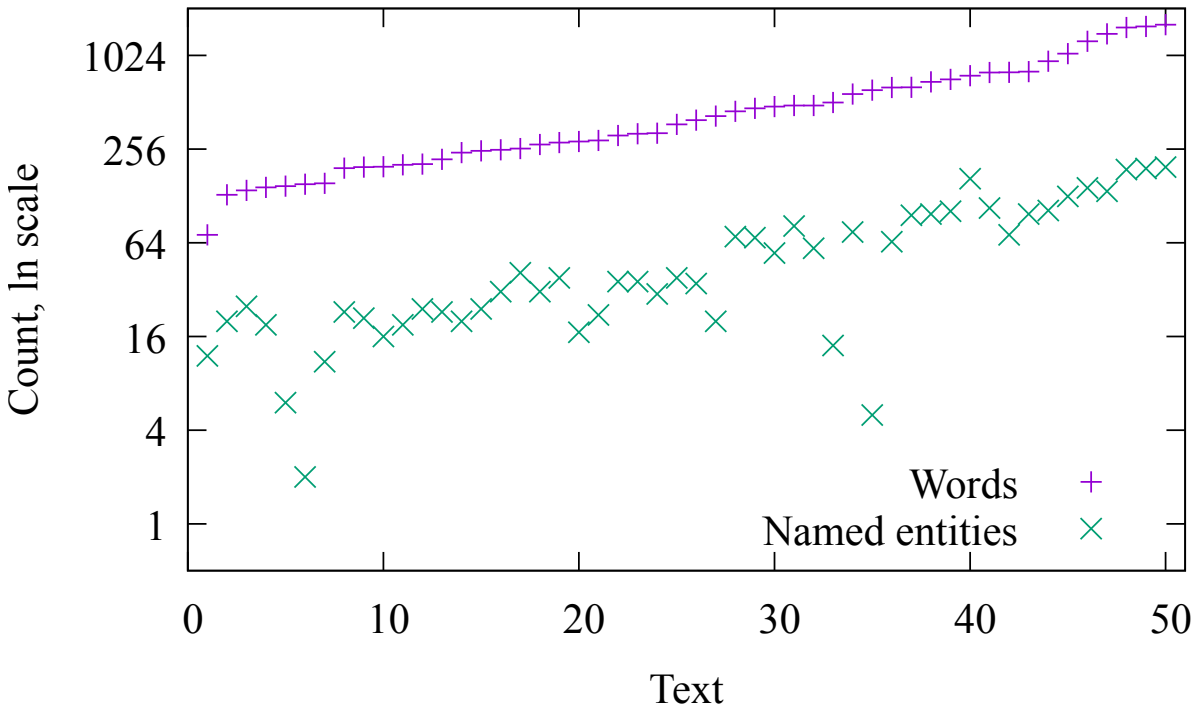

Fig. 1. Words and NE count for texts in the corpus

Рис. 1. Структура корпуса текстов

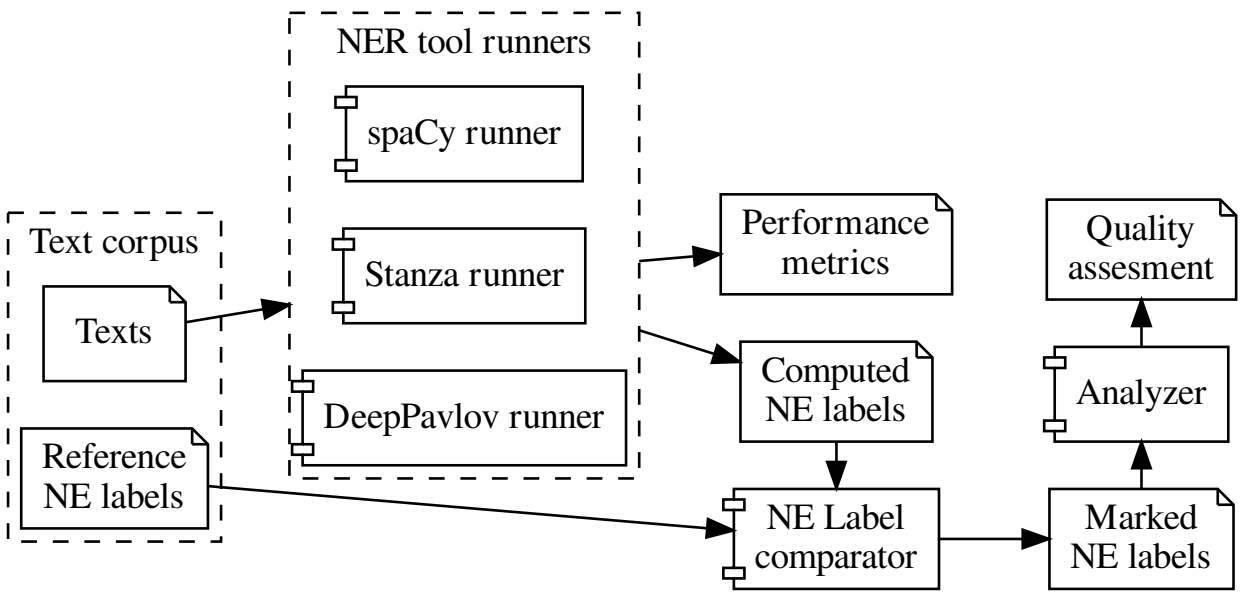

Fig. 2. Test testbed components and data they use for operation

Рис. 2. Схема процесса тестирования

соответствующее приложение, которое обеспечивает унифицированный интерфейс: принимает

текст в качестве входных данных и предоставляет распознанные именованные объекты в общем формате.

Помимо типов NE, методы инструментов предоставляют показатели производительности, которые позволяют отслеживать время выполнения распознавания именованных объектов. Для уменьшения влияния вариабельности отдельного эксперимента, каждый запуск инструмента повторялся 10 раз. В качестве результата взяты среднее арифметическое и медиана времени выполнения.

Следующим важным компонентом испытательного стенда является анализатор распознавания категорий именованных сущностей. Он берет категории NE, вычисленные инструментом, и сравнивает их с эталонными. Результатом работы компонента является список NE, в котором указывается, является ли вычисленная категория NE правильной или нет. Последний шаг — это расчет F-меры. Оценка рассчитывается для всех именованных объектов и для каждой поддерживаемой категории.

Помимо компонентов, показанных на рис. 2, существует сценарий запуска, который способен выполнять все остальные компоненты. Во время одного выполнения пользователь может выбрать инструмент, который он хочет запустить, и указать местоположение исследуемого корпуса текстов. Категория MISC распознаётся только Stanza.

6.4. Результаты экспериментов

Table 7. Results of experiment **Таблица 7.** Результаты экспериментов

	Stanza с MISC	Stanza без MISC	spaCy	DeepPavlov
F-мера, %	82	87.1	79.5	89
F-мера PER, %	94.1	94.1	86.1	93.8
F-мера ORG, %	75.7	75.7	66	79.8
F-мера LOC, %	95.9	95.9	88.2	94
F-мера MISC, %	65.2	–	–	–
Среднее время исполнения, сек.	138.376	138.376	80.604	79.938
Медианное время исполнения, сек.	138.042	138.042	78.81	79.36

Эксперименты проводились на персональном компьютере, оснащенном процессором Intel® Core™ i5-9300H, работающим под управлением операционной системы Ubuntu 20.04. В ходе экспериментов использовались последние выпущенные версии инструментов: Stanza 1.3, spaCy 3.2 и DeepPavlov 3.7.9. Все инструменты используют только процессор для работы нейронной сети.

Результаты экспериментов представлены в таблице 7. Лучшие результаты можно увидеть у DeepPavlov с F-мерой 89 %, Stanza и spaCy следуют с 82 % и 79,5 % соответственно. Если не учитывать MISC, Stanza обеспечивает F-меру 87,1 %, сокращая разрыв между DeepPavlov. Если учитывать F-меру и время выполнения, DeepPavlov является лучшим инструментом для собранного корпуса. Все инструменты дают хорошие результаты для именованных сущностей категорий PER и LOC, но плохие результаты для категории ORG. Категория MISC также плохо распознается Stanza.

7. Заключение

Самый большой прогресс в области NER достигнут с помощью моделей, основанных на глубоком обучении и зависящих от объёмных эталонных наборов данных. Это относится как отдельным типам задач в этой области, так и к обработке текстов на национальных языках. Создание размеченных корпусов данных помогает решить эту проблему, но является очень дорогостоящим процессом, требующим постоянного обновления в связи с изменяющимся состоянием сфер жизни.

Одним из перспективных направлений исследований в области NER является развитие методов на основе обучения без учителя или на основе правил. Возможной базой предобработки текста для таких методов могут служить интенсивно развивающиеся модели языков в существующих инструментах NLP. Такой подход может решить проблему больших вычислительных ресурсов, которые

требуются для автономных систем NER. Кроме того, использование сложных лексических параметров текста и баз правил позволит лучше понять и объяснить процессы и результаты решения поставленных задач.

References

[1] R. Grishman and B. Sundheim, "Message understanding conference-6: A brief history", in *Proceedings of the 16th International Conference on Computational Linguistics (COLING 96), Copenhagen, August 1996*, 1996, pp. 466–471.

[2] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification", *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[3] R. Sharnagat, "Named entity recognition: A literature survey", *Center For Indian Language Technology*, pp. 1–27, 2014.

[4] J. Li, A. Sun, J. Han, and C. Li, "A Survey on Deep Learning for Named Entity Recognition", *IEEE Transactions on Knowledge & Data Engineering*, vol. 34, no. 1, pp. 50–70, 2022.

[5] G. Popovski, B. K. Seljak, and T. Eftimov, "A survey of named-entity recognition methods for food information extraction", *IEEE Access*, vol. 8, pp. 31 586–31 594, 2020.

[6] J. Piskorski, B. Babych, Z. Kancheva, O. Kanishcheva, M. Lebedeva, M. Marcińczuk, P. Nakov, P. Osenova, L. Pivovarova, S. Pollak, *et al.*, "Slav-NER: the 3rd Cross-lingual Challenge on Recognition, Normalization, Classification, and Linking of Named Entities across Slavic Languages", in *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, 2021, pp. 122–133.

[7] A. Starostin, V. Bocharov, S. Alexeeva, A. Bodrova, A. Chuchunkov, *et al.*, "FactRuEval 2016: Evaluation of named entity recognition and fact extraction systems for Russian", in *Computational Linguistics and Intellectual Technologies*, 2016, pp. 702–720.

[8] Y. Jiang, C. Hu, T. Xiao, C. Zhang, and J. Zhu, "Improved differentiable architecture search for language modeling and named entity recognition", in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3585–3590.

[9] R. Speck and A.-C. Ngonga Ngomo, "Ensemble learning for named entity recognition", in *International semantic web conference*, Springer, 2014, pp. 519–534.

[10] A. Ghaddar and P. Langlais, "Robust Lexical Features for Improved Neural Network Named-Entity Recognition", in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 1896–1907.

[11] A. Akbik, T. Bergmann, and R. Vollgraf, "Pooled contextualized embeddings for named entity recognition", in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 724–728.

[12] M. Riedl and S. Padó, "A named entity recognition shootout for german", in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 120–125.

[13] P. H. L. de Araujo, T. E. de Campos, R. R. de Oliveira, M. Stauffer, S. Couto, and P. Bermejo, "Lener-br: a dataset for named entity recognition in brazilian legal text", in *International Conference on Computational Processing of the Portuguese Language*, Springer, 2018, pp. 313–323.

[14] Y. Luo, F. Xiao, and H. Zhao, "Hierarchical contextualized representation for named entity recognition", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 8441–8448.

[15] N. Eligüzel, C. Çetinkaya, and T. Dereli, "Application of named entity recognition on tweets during earthquake disaster: a deep learning-based approach", *Soft Computing*, vol. 26, no. 1, pp. 395–421, 2022.

[16] M. Y. Arkhipov, M. S. Burtsev, *et al.*, "Application of a Hybrid Bi-LSTM-CRF model to the task of Russian Named Entity Recognition", in *Conference on Artificial Intelligence and Natural Language*, Springer, 2017, pp. 91–103.

[17] R. Hvingelby, A. B. Pauli, M. Barrett, C. Rosted, L. M. Lidegaard, and A. Søgaard, "DaNE: A named entity resource for danish", in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 4597–4604.

[18] M. Arkhipov, M. Trofimova, Y. Kuratov, and A. Sorokin, "Tuning multilingual transformers for named entity recognition on slavic languages", *BSNLP'2019*, p. 89, 2019.

[19] J. Piskorski, L. Laskova, M. Marcińczuk, L. Pivovarova, P. Přibáň, J. Steinberger, and R. Yangarber, "The Second Cross-Lingual Challenge on Recognition, Normalization, Classification, and Linking of Named Entities across Slavic Languages", in *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*, 2019, pp. 63–74.

[20] C. Helwe, G. Dib, M. Shamas, and S. Elbassuoni, "A semi-supervised BERT approach for Arabic named entity recognition", in *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, 2020, pp. 49–57.

[21] C. Liang, Y. Yu, H. Jiang, S. Er, R. Wang, T. Zhao, and C. Zhang, "Bond: BERT-assisted open-domain named entity recognition with distant supervision", in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1054–1064.

[22] E. Oliveira, G. Dias, J. Lima, and J. Pirovani, "Using Named Entities for Recognizing Family Relationships", in *Anais do IX Symposium on Knowledge Discovery, Mining and Learning*, SBC, 2021, pp. 24–32.

[23] R. Yeniterzi, G. Tür, and K. Oflazer, "Turkish named-entity recognition", in *Turkish Natural Language Processing*, Springer, 2018, pp. 115–132.

[24] T. Ruokolainen, P. Kauppinen, M. Silfverberg, and K. Lindén, "A Finnish news corpus for named entity recognition", *Language Resources and Evaluation*, vol. 54, no. 1, pp. 247–272, 2020.

[25] Y. Fu, N. Lin, Z. Yang, and S. Jiang, "Towards Malay named entity recognition: an open-source dataset and a multi-task framework", *Connection Science*, pp. 1–23, 2022.

[26] M. Ju, M. Miwa, and S. Ananiadou, "A neural layered model for nested named entity recognition", in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1446–1459.

[27] C. Xia, C. Zhang, T. Yang, Y. Li, N. Du, X. Wu, W. Fan, F. Ma, and P. Yu, "Multi-grained named entity recognition", in *57th Annual Meeting of the Association for Computational Linguistics, ACL 2019*, Association for Computational Linguistics (ACL), 2020, pp. 1430–1440.

[28] J. Wan, D. Ru, W. Zhang, and Y. Yu, "Nested Named Entity Recognition with Span-level Graphs", in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022, pp. 892–903.

[29] J. Yu, B. Bohnet, and M. Poesio, "Named Entity Recognition as Dependency Parsing", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 6470–6476.

[30] M. G. Sohrab and M. Miwa, "Deep exhaustive model for nested named entity recognition", in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2843–2849.

[31] J. Li, D. Ye, and S. Shang, "Adversarial Transfer for Named Entity Boundary Detection with Pointer Networks", in *IJCAI*, 2019, pp. 5053–5059.

[32] D. Bareket and R. Tsarfaty, "Neural modeling for named entities and morphology (nemo^2)", *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 909–928, 2021.

[33] S. H. Jeon and S. Cho, "Edge Weight Updating Neural Network for Named Entity Normalization", *Neural Processing Letters*, pp. 1–22, 2022.

[34] D. Zhou and T. Liu, "Joint model of biomedical entity recognition and normalization labels based on self-attention", in *Second International Symposium on Computer Technology and Information Science (ISCTIS 2022)*, SPIE, vol. 12474, 2022, pp. 461–466.

[35] A. Fritzler, V. Logacheva, and M. Kretov, "Few-shot classification in named entity recognition task", in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 993–1000.

[36] J. R. Finkel and C. D. Manning, "Nested named entity recognition", in *Proceedings of the 2009 conference on empirical methods in natural language processing*, 2009, pp. 141–150.

[37] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations. arXiv preprint arXiv: 180205365", in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, 2018, pp. 2227–2237.

[38] A. Ghaddar, P. Langlais, A. Rashid, and M. Rezagholizadeh, "Context-aware adversarial training for name regularity bias in named entity recognition", *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 586–604, 2021.

[39] Y. Nie, Y. Tian, Y. Song, X. Ao, and X. Wan, "Improving Named Entity Recognition with Attentive Ensemble of Syntactic Information", in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 4231–4245.

[40] G. Aguilar, S. Maharjan, A. P. López-Monroy, and T. Solorio, "A Multi-task Approach for Named Entity Recognition in Social Media Data", *W-NUT 2017*, p. 148, 2017.

[41] Q. Zhang, J. Fu, X. Liu, and X. Huang, "Adaptive Co-attention Network for Named Entity Recognition in Tweets", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018, pp. 5674–5681.

[42] A. Miranda-Escalada, E. Farré-Maduell, S. Lima-López, L. Gascó, V. Briva-Iglesias, M. Agüero-Torales, and M. Krallinger, "The profner shared task on automatic recognition of occupation mentions in social media: systems, evaluation, guidelines, embeddings and corpora", in *Proceedings of the Sixth Social Media Mining for Health (# SMM4H) Workshop and Shared Task*, 2021, pp. 13–20.

[43] M. Asgari-Chenaghlu, M. R. Feizi-Derakhshi, L. Farzinvash, M. Balafar, and C. Motamed, "CWI: A multimodal deep learning approach for named entity recognition from social media using character, word and image features", *Neural Computing and Applications*, pp. 1–18, 2021.

[44] J. Yu, J. Jiang, L. Yang, and R. Xia, "Improving Multimodal Named Entity Recognition via Entity Span Detection with Unified Multimodal Transformer", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3342–3352.

[45] M. Asgari-Bidhendi, B. Janfada, O. Roshani Talab, and B. Minaei-Bidgoli, "ParsNER-Social: A Corpus for Named Entity Recognition in Persian Social Media Texts", *Journal of AI and Data Mining*, vol. 9, no. 2, pp. 181–192, 2021.

[46] R. Priyadharshini, B. R. Chakravarthi, M. Vegupatti, and J. P. McCrae, "Named entity recognition for code-mixed Indian corpus using meta embedding", in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, IEEE, 2020, pp. 68–72.

[47] M. C. Phan and A. Sun, "Collective named entity recognition in user comments via parameterized label propagation", *Journal of the Association for Information Science and Technology*, vol. 71, no. 5, pp. 568–577, 2020.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need", *Advances in neural information processing systems*, vol. 30, pp. 5998–6008, 2017.

[49] X. Wang, Y. Zhang, X. Ren, Y. Zhang, M. Zitnik, J. Shang, C. Langlotz, and J. Han, "Cross-type biomedical named entity recognition with deep multi-task learning", *Bioinformatics*, vol. 35, no. 10, pp. 1745–1752, 2019.

[50] W. Yoon, C. H. So, J. Lee, and J. Kang, "Collabonet: collaboration of deep neural networks for biomedical named entity recognition", *BMC bioinformatics*, vol. 20, no. 10, pp. 55–65, 2019.

[51] L. Weber, M. Sänger, J. Münchmeyer, M. Habibi, U. Leser, and A. Akbik, "HunFlair: an easy-to-use tool for state-of-the-art biomedical named entity recognition", *Bioinformatics*, vol. 37, no. 17, pp. 2792–2794, 2021.

[52] D. Kim, J. Lee, C. H. So, H. Jeon, M. Jeong, Y. Choi, W. Yoon, M. Sung, and J. Kang, "A neural named entity recognition and multi-type normalization tool for biomedical text mining", *IEEE Access*, vol. 7, pp. 73 729–73 740, 2019.

[53] Z. Miftahutdinov, I. Alimova, and E. Tutubalina, "On biomedical named entity recognition: experiments in interlingual transfer for clinical and social media texts", in *European Conference on Information Retrieval*, Springer, 2020, pp. 281–288.

[54] R. Catelli, F. Gargiulo, V. Casola, G. De Pietro, H. Fujita, and M. Esposito, "Crosslingual named entity recognition for clinical de-identification applied to a COVID-19 Italian data set", *Applied Soft Computing*, vol. 97, p. 106 779, 2020.

[55] L. Luo, Z. Yang, P. Yang, Y. Zhang, L. Wang, H. Lin, and J. Wang, "An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition", *Bioinformatics*, vol. 34, no. 8, pp. 1381–1388, 2018.

[56] W. Hemati and A. Mehler, "LSTMVoter: chemical named entity recognition using a conglomerate of sequence labeling tools", *Journal of cheminformatics*, vol. 11, no. 1, pp. 1–7, 2019.

[57] Z. Hong, R. Tchoua, K. Chard, and I. Foster, "SciNER: extracting named entities from scientific literature", in *International Conference on Computational Science*, Springer, 2020, pp. 308–321.

[58] L. Weston, V. Tshitoyan, J. Dagdelen, O. Kononova, A. Trewartha, K. A. Persson, G. Ceder, and A. Jain, "Named entity recognition and normalization applied to large-scale information extraction from the materials science literature", *Journal of chemical information and modeling*, vol. 59, no. 9, pp. 3692–3702, 2019.

[59] A. Kumar and B. Starly, ""FabNER": information extraction from manufacturing process science domain literature using named entity recognition", *Journal of Intelligent Manufacturing*, vol. 33, no. 8, pp. 2393–2407, 2022.

[60] S. Moon, G. Lee, S. Chi, and H. Oh, "Automated construction specification review with named entity recognition using natural language processing", *Journal of Construction Engineering and Management*, vol. 147, no. 1, p. 04 020 147, 2021.

[61] M. H. Syed and S.-T. Chung, "MenuNER: Domain-adapted BERT based NER approach for a domain with limited dataset and its application to food menu domain", *Applied Sciences*, vol. 11, no. 13, p. 6007, 2021.

[62] G. Popovski, S. Kochev, B. Korousic-Seljak, and T. Eftimov, "FoodIE: A Rule-based Named-entity Recognition Method for Food Information Extraction.", in *ICPRAM*, 2019, pp. 915–922.

[63] F.-L. Li, H. Chen, G. Xu, T. Qiu, F. Ji, J. Zhang, and H. Chen, "AliMeKG: Domain knowledge graph construction and application in e-commerce", in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 2581–2588.

[64] N. Perera, T. T. L. Nguyen, M. Dehmer, and F. Emmert-Streib, "Comparison of text mining models for food and dietary constituent named-entity recognition", *Machine Learning and Knowledge Extraction*, vol. 4, no. 1, pp. 254–275, 2022.

[65] T. Eftimov, B. Koroušić Seljak, and P. Korošec, "A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations", *PloS one*, vol. 12, no. 6, e0179488, 2017.

[66] Z. Liu, M. Luo, H. Yang, and X. Liu, "Named entity recognition for the horticultural domain", in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1631, 2020, p. 012 016.

[67] G. Kim, C. Lee, J. Jo, and H. Lim, "Automatic extraction of named entities of cyber threats using a deep Bi-LSTM-CRF network", *International journal of machine learning and cybernetics*, vol. 11, no. 10, pp. 2341–2355, 2020.

[68] S. Zhou, J. Liu, X. Zhong, and W. Zhao, "Named Entity Recognition Using BERT with Whole World Masking in Cybersecurity Domain", in *2021 IEEE 6th International Conference on Big Data Analytics (ICBDA)*, IEEE, 2021, pp. 316–320.

[69] M. Tikhomirov, N. Loukachevitch, A. Sirotina, and B. Dobrov, "Using BERT and augmentation in named entity recognition for cybersecurity domain", in *International Conference on Applications of Natural Language to Information Systems*, Springer, 2020, pp. 16–24.

[70] T. W. T. Au, I. J. Cox, and V. Lampos, "E-NER–An Annotated Named Entity Recognition Corpus of Legal Text", *arXiv preprint arXiv:2212.09306*, 2022.

[71] C. Cetindag, B. Yaziciouglu, and A. Koc, "Named-entity recognition in Turkish legal texts", *Natural Language Engineering*, pp. 1–28, 2022.

[72] A. Brandsen, S. Verberne, K. Lambers, and M. Wansleeben, "Can BERT Dig It? Named Entity Recognition for Information Retrieval in the Archaeology Domain", *Journal on Computing and Cultural Heritage (JOCCH)*, vol. 15, no. 3, pp. 1–18, 2022.

[73] E. Kogkitsidou and P. Gambette, "Normalisation of 16th and 17th century texts in French and geographical named entity recognition", in *Proceedings of the 4th ACM SIGSPATIAL Workshop on Geospatial Humanities*, 2020, pp. 28–34.

[74] D. Alexander and A. P. de Vries, "" This research is funded by...": Named Entity Recognition of Financial Information in Research Papers", in *Proceedings of the 11th International Workshop on Bibliometric-enhanced Information Retrieval co-located with 43rd ECIR*, Sl: CEUR, 2021, pp. 102–110.

[75] J. Li, S. Shang, and L. Shao, "Metaner: Named entity recognition with meta-learning", in *Proceedings of The Web Conference 2020*, 2020, pp. 429–440.

[76] B. Nie, C. Li, and H. Wang, "KA-NER: Knowledge Augmented Named Entity Recognition", in *China Conference on Knowledge Graph and Semantic Computing*, Springer, 2021, pp. 60–75.

[77] B.-S. Lin, J.-H. Chen, and T.-H. Chang, "NERVE at ROCLING 2022 shared task: a comparison of three named entity recognition frameworks based on language model and lexicon approach", in *Proceedings of the 34th Conference on Computational Linguistics and Speech Processing (ROCLING 2022)*, 2022, pp. 343–349.

[78] T. Isazawa and J. M. Cole, "Single Model for Organic and Inorganic Chemical Named Entity Recognition in ChemDataExtractor", *Journal of Chemical Information and Modeling*, vol. 62, no. 5, pp. 1207–1213, 2022.

[79] B. Singh, A. Marathe, A. A. Rizvi, and A. R. Joshi, "Retaining Named Entities for Headline Generation", in *Inventive Computation and Information Technologies*, Springer, 2021, pp. 221–234.

[80] Y. Ji, C. Tong, J. Liang, X. Yang, Z. Zhao, and X. Wang, "A deep learning method for named entity recognition in bidding document", in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1168, 2019, p. 032 076.

[81] S. M. Makeev, S. V. SHekshuev, M. G. Petrov, and S. D. Zyuzin, "Modul' obrabotki estestvennogo yazyka na osnove obuchennoj modeli nejronnoj seti s mekhanizmom poiska imenovannyh sushchnostej", *Izvestiya Tul'skogo gosudarstvennogo universiteta. Tekhnicheskie nauki*, no. 9, pp. 56–64, 2022.

[82] K. Krasnashchok and S. Jouili, "Improving topic quality by promoting named entities in topic modeling", in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 247–253.

[83] A. Kumar and B. Starly, ""FabNER": information extraction from manufacturing process science domain literature using named entity recognition", *Journal of Intelligent Manufacturing*, pp. 1–15, 2021.

[84] A. Siekmeier, W. Lee, H. Kwon, and J.-H. Lee, "Tag Assisted Neural Machine Translation of Film Subtitles", in *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, 2021, pp. 255–262.

[85] J. Ding, H. Sun, X. Wang, and X. Liu, "Entity-level sentiment analysis of issue comments", in *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*, 2018, pp. 7–13.

[86] J. Yu, J. Jiang, and R. Xia, "Entity-sensitive attention and fusion network for entity-level multimodal sentiment classification", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 429–439, 2019.

[87] Z. A. Guven and M. O. Unalir, "Improving the BERT Model with Proposed Named Entity Recognition Method for Question Answering", in *2021 6th International Conference on Computer Science and Engineering (UBMK)*, IEEE, 2021, pp. 204–208.

[88] B. Kleinberg, M. Mozes, A. Arntz, and B. Verschuere, "Using named entities for computer-automated verbal deception detection", *Journal of forensic sciences*, vol. 63, no. 3, pp. 714–723, 2018.

[89] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, *spaCy: Industrial-strength Natural Language Processing in Python, Zenodo, 2020.*

[90] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, "Stanza: A Python Natural Language Processing Toolkit for Many Human Languages", in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020. [Online]. Available: https://nlp.stanford.edu/pubs/qi2020stanza.pdf.

[91] J. Nothman, N. Ringland, W. Radford, T. Murphy, and J. R. Curran, "Learning multilingual named entity recognition from Wikipedia", *Artificial Intelligence*, vol. 194, pp. 151–175, 2013.

[92] M. S. Burtsev, A. V. Seliverstov, R. Airapetyan, M. Arkhipov, D. Baymurzina, N. Bushkov, O. Gureenkova, T. Khakhulin, Y. Kuratov, D. Kuznetsov, *et al.*, "DeepPavlov: Open-Source Library for Dialogue Systems.", in *ACL (4)*, 2018, pp. 122–127.

[93] V. Mozharova and N. Loukachevitch, "Two-stage approach in Russian named entity recognition", in *2016 International FRUCT Conference on Intelligence, Social Media and Web (ISMW FRUCT)*, IEEE, 2016, pp. 1–6.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 1, 2023
journal homepage: www.mais-journal.ru

THEORY OF DATA

Annotation of Text Corpora by Sentiment and Presence of Irony within a Project of Citizen Science

I. V. Paramonov[1], A. Y. Poletaev[1]

DOI: 10.18255/1818-1015-2023-1-86-100

[1]P. G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 68T50
Research article
Full text in Russian

Received February 3, 2023
After revision February 24, 2023
Accepted February 27, 2023

The paper is devoted to construction of a sentence corpus annotated by the general sentiment into 4 classes (positive, negative, neutral, and mixed), a corpus of phrasemes annotated by the sentiment into 3 classes (positive, negative, and neutral), and a corpus of sentences annotated by the presence or absence of irony. The annotation was done by volunteers within the project "Prepare texts for algorithms" on the portal "People of science".
The existing knowledge on the domain regarding each task was the basis to develop guidelines for annotators. A technique of statistical analysis of the annotation result based on the distributions and agreement measures of the annotations performed by various annotators was also developed. For the annotation of sentences by irony and phrasemes by the sentiment the agreement measures were rather high (the full agreement rate of 0.60–0.99), whereas for the annotation of sentences by the general sentiment the agreement was low (the full agreement rate of 0.40), presumably, due to the higher complexity of the task. It was also shown that the results of automatic algorithms of detecting the sentiment of sentences improved by 12–13 % when using a corpus for which all the annotators (from 3 till 5) had the agreement, in comparison with a corpus annotated by only one volunteer.

Keywords: sentiment analysis; text corpus; statistical analysis; agreement measures; citizen science

INFORMATION ABOUT THE AUTHORS

Ilya Vyacheslavovich Paramonov (correspondence author) | orcid.org/0000-0003-3984-8423. E-mail: ilya.paramonov@fruct.org
PhD, associate professor.

Anatoliy Yurievich Poletaev | orcid.org/0000-0003-0116-4739. E-mail: anatoliy-poletaev@mail.ru
post-graduate student.

Funding: The research was performed within the YarSU citizen science project No. CS-02/2022.

For citation: I. V. Paramonov and A. Y. Poletaev, "Annotation of Text Corpora by Sentiment and Presence of Irony within a Project of Citizen Science", *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 86-100, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 1, 2023
сайт журнала: www.mais-journal.ru

THEORY OF DATA

Разметка корпусов текстов по тональности и наличию иронии в рамках проекта гражданской науки

И. В. Парамонов[1], А. Ю. Полетаев[1]

DOI: 10.18255/1818-1015-2023-1-86-100

[1]Ярославский государственный университет им. П. Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 004.912
Научная статья
Полный текст на русском языке

Получена 3 февраля 2023 г.
После доработки 24 февраля 2023 г.
Принята к публикации 27 февраля 2023 г.

Статья посвящена построению корпуса предложений, размеченных по общей тональности на 4 класса (положительный, отрицательный, нейтральный, смешанный), корпуса фразеологизмов, размеченных по тональности на 3 класса (положительный, отрицательный, нейтральный), и корпуса предложений, размеченных по наличию или отсутствию иронии. Разметку проводили волонтёры в рамках проекта «Готовим тексты алгоритмам» на портале «Люди науки».

На основе имеющихся знаний о предметной области для каждой из задач были составлены инструкции для разметчиков. Также была выработана методика статистической обработки результатов разметки, основанная на анализе распределений и показателей согласия оценок, выставленных разными разметчиками. Для разметки предложений по наличию иронии и фразеологизмов по тональности показатели согласия оказались достаточно высокими (доля полного совпадения 0.60–0.99), при разметке предложений по общей тональности согласие оказалось слабым (доля полного совпадения 0.40), по-видимому, из-за более высокой сложности задачи. Также было показано, что результаты работы автоматических алгоритмов анализа тональности предложений улучшаются на 12–13 % при использовании корпуса, относительно предложений которого сошлись мнения всех разметчиков (3–5 человек), по сравнению с корпусом с разметкой только одним волонтёром.

Ключевые слова: анализ тональности; текстовый корпус; статистический анализ; показатели согласия; гражданская наука

ИНФОРМАЦИЯ ОБ АВТОРАХ

Илья Вячеславович Парамонов
автор для корреспонденции
orcid.org/0000-0003-3984-8423. E-mail: ilya.paramonov@fruct.org
канд. физ.-мат. наук, доцент.

Анатолий Юрьевич Полетаев
orcid.org/0000-0003-0116-4739. E-mail: anatoliy-poletaev@mail.ru
аспирант.

Финансирование: Исследование выполнено в рамках проекта гражданской науки ЯрГУ № CS-02/2022.

Для цитирования: I. V. Paramonov and A. Y. Poletaev, “Annotation of Text Corpora by Sentiment and Presence of Irony within a Project of Citizen Science”, *Modeling and analysis of information systems*, vol. 30, no. 1, pp. 86-100, 2023.

Введение

Построение размеченных текстовых корпусов является актуальной задачей компьютерной лингвистики, поскольку предоставляет исходные данные практически для любых исследований в данной области. Решение данной задачи является чрезвычайно трудозатратным, так как достижение высокого качества построенных корпусов требует большого количества работы, выполняемой вручную. Поскольку критерии разметки часто оказываются трудноформализуемыми (например, в случае разметки текста по тональности), желательным является выполнение разметки экспертами в предметной области, что ещё больше увеличивает стоимость выполнения данного вида работ.

В связи с высокой трудоёмкостью исследователи часто прибегают к автоматической или полуавтоматической разметке текстов (например, разметке текстовых отзывов на товары по их рейтингу, задаваемому пользователями на маркетплейсах), а также привлечению неквалифицированных разметчиков-волонтёров. В обоих случаях качество корпусов будет ниже, чем в случае ручной разметки экспертами, однако построенные таким образом корпуса всё же могут быть полезны в различных исследованиях после агрегации собранных данных и хотя бы избирательной их верификации [1].

Не существует общей методики составления инструкций по разметке текстовых корпусов для волонтёров, а также обработки результатов разметки. Составление инструкций — достаточно сложная задача, т. к. они, с одной стороны, должны быть достаточно полными, чтобы неквалифицированные разметчики могли в подавляющем большинстве случаев ориентироваться на них, а с другой стороны, не должны быть слишком сложными и объёмными, чтобы избежать сложностей с их пониманием. От качества методики обработки результатов разметки напрямую зависит применимость полученного корпуса для решения задач компьютерной лингвистики [2].

Настоящая работа посвящена построению корпуса предложений, размеченных по общей тональности на 4 класса (положительный, отрицательный, нейтральный, смешанный), корпуса фразеологизмов, размеченных по тональности на 3 класса (положительный, отрицательный, нейтральный), и корпуса предложений, размеченных по наличию или отсутствию иронии. Разметку проводили волонтёры, набранные в рамках проекта «Готовим тексты алгоритмам» на портале «Люди науки»[1].

1. Обзор связанных работ

В современной научной литературе крайне мало исследований, посвящённых разметке текстовых корпусов и агрегации результатов этой разметки. В большинстве работ, где идёт речь о построении размеченных текстовых корпусов, авторы очень поверхностно описывают или не описывают вообще, как именно они собирались. Можно выделить два основных аспекта, связанных с данной задачей: разработка инструкций по разметке и собственно методика разметки и агрегации её результатов.

Нужно отметить, что существует несколько общих методик агрегации экспертных оценок, например, метод Делфи, однако из-за достаточно высокой трудоёмкости они слабо применимы для агрегации оценок большого числа предложений, выставленных большим числом разметчиков.

Одной из наиболее значимых работ в области построения инструкций для разметки текстов по тональности является статья [3]. В ней отмечается потенциальная сложность задачи, вызванная нечётким определением самого понятия тональности, приводится классификация предложений, сложных для разметки, и предлагаются два подхода к выработке инструкций для разметчиков, предположительно позволяющие повысить точность результатов в сложных случаях. В первом случае используется опросник, содержащий утверждения, характеризующие те или иные классы тональности предложений, а также примеры типов предложений, относящихся к заданному классу (восхищение, поддержка, симпатия и т. п.). Во втором случае разметчикам предлагается отвечать

[1] https://citizen-science.ru/projects/gotovim-teksty-algoritmam.html

на вопросы относительно эмоционального состояния автора предложения, объекта, по отношению к которому выражена тональность, и характера воздействия предложения на большинство людей. Отмечается, что среди этих двух подходов нет наилучшего, и эффективность каждого из них может определяться решаемой задачей и квалификацией разметчиков.

В другой работе того же автора [4] рассматривается задача разметки твитов по отношению автора к конкретным персоналиям или явлениям (за, против, другое) и о том, высказывается ли мнение об объекте напрямую. Каждый твит рассматривался как минимум восемью разметчиками, что является очень большим значением среди всех существующих работ в данной области. Всего было собрано 5412 твитов; 5 % твитов были изначально размечены их авторами, и если разметчик ошибался при разметке такого твита, он получал уведомление об этом, а если он ошибался более чем на 30 % таких твитов в совокупности, он исключался из работы как недобросовестный. При агрегации результатов брались только такие оценки, с которыми были согласны как минимум 60 % работавших с ними разметчиков. Степень согласия составила примерно 81.85 % для задачи определения отношения автора к персоналии или явлению и 68.9 % для задачи определения того, выражено ли мнение об объекте напрямую. Степень согласия определялась как среднее мер согласия по каждому твиту, которая, в свою очередь, вычислялась как отношение количества разметчиков, давших наиболее распространённый ответ для данного твита, к общему количеству разметчиков этого твита.

В статье [5] используется подход из работы [3] с некоторыми изменениями. В частности, рассматриваются только предложения из 5–15 слов. Кроме того, идея проверки следования разметчика инструкциям, решаемая в оригинальной работе с использованием вопросов с известными ответами, о которых разметчик заранее не знал, заменена выдачей участникам перед началом работы размеченного экспертами набора предложений в качестве примера. Каждое предложение размечалось минимум двумя разметчиками. В случае их согласия третий не привлекался, в случае несогласия — привлекался. Если и третий не давал прийти к решению, привлекались ещё двое. Данные некоторых разметчиков были исключены из рассмотрения, поскольку они допускали слишком много ошибок на первом этапе. Всего было размечено 15744 предложения. Для оценки согласия использовалась альфа Криппендорфа [6], показывающая насколько согласие между разметчиками близко к полному. Её значение получилось равным 0.66, что может говорить о существенном согласии. Также приведены 4 примера сложных для анализа предложений, сходных с описанными в работе [3].

Сходная методика используется в статье [7]: разметка производилась двумя экспертами; в случае, когда оценки не сходились, решение принимал третий разметчик. Всего было размечено 17572 предложения на китайском языке. Для оценки согласия использовалась каппа Коэна [8]. Её значение оказалось равным 0.71–0.73, что считается удовлетворительным.

В другой работе также для китайского языка [9] размечался корпус отзывов на рестораны по различным аспектам (качество еды, обслуживания, цена и т. д.). Для каждого аспекта разметчикам предлагалось ответить только на один вопрос — об отношении автора отзыва к аспекту (положительном, отрицательном или нейтральном). Каждое предложение размечалось двумя независимыми разметчиками, прошедшими краткое обучение; далее проводилась валидация результатов их разметки третьим разметчиком (если оценки сошлись) или экспертом (в противном случае). Спорные случаи также обрабатывались экспертом. В итоге было размечено 46730 предложений. Характеристики согласия не вычислялись.

В работе [10] производилась разметка корпуса твитов, относящихся к определённым брендам. Каждый твит размечался тремя разметчиками, которые должны были указать, какие эмоции из заданного набора (доверие, счастье, грусть и т. д.) присутствуют в заданном твите. Для оценки согласия всех разметчиков по категориям использовалась каппа Флейса [11], для оценки попарного согласия между разметчиками — каппа Коэна. Обе эти метрики характеризуют долю совпавших оценок разметчиков, которая не может быть объяснена случайным совпадением мнений. Значения метрик

составили 0.372 и 0.354 соответственно, что говорит об умеренном согласии разметчиков. Анализ причин несогласия разметчиков не проводился.

В работе [12] рассматривается задача разметки по тональности корпуса постов из социальной сети «ВКонтакте» по социально-политической тематике с вариантами ответа: «положительное», «отрицательное», «не тональное», «сомнительное», «речевой акт» (поздравления, технические сообщения и т. д.). Размечались только посты, содержащие от 10 до 800 символов. К разметке были привлечены 6 экспертов в области лингвистики, которым были выданы соответствующие инструкции. В итоге были размечены 31185 постов, каждый пост размечался тремя разметчиками. Для оценки согласия использовалась каппа Флейса, величина которой составила 0.58.

Из рассмотренных работ видно, что в каждом конкретном случае разметка корпусов осуществлялась по-разному и использовались различные метрики для оценки согласия. Есть и общие черты: практически во всех работах каждый текстовый фрагмент размечается минимум тремя разметчиками (в некоторых случаях ограничиваются даже двумя оценками, если разметчики согласны между собой); используются некоторые инструкции для разметчиков, весьма схожие в разных работах; предпринимаются действия по очистке размеченного корпуса от сомнительных результатов разметки вплоть до удаления результатов тех разметчиков, которые представляются недобросовестными. В большинстве перечисленных работ основным показателем качества разметки считаются метрики согласия. Возможно, это связано с тем, что другие методы анализа, например, верификация экспертами, слишком трудоёмки. Однако степень согласия между разметчиками оказывается достаточно низкой, что, по-видимому, обусловлено нечёткостью понятия тональности и субъективным его восприятием разными людьми.

2. Методика разметки корпусов

В качестве исходного корпуса для разметки предложений по тональности и по наличию иронии был использован OpenCorpora[2] — открытый корпус предложений из новостных и публицистических текстов на русском языке, относящихся к различным предметным областям. Для задачи разметки предложений по тональности использовались только предложения из семи и более слов. Список фразеологизмов и их значений для разметки по тональности был взят из Викисловаря[3].

Перед разметчиками ставилась задача оценить с точки зрения тональности или наличия иронии предложения или фразеологизмы из индивидуального набора. Наборы формировались автоматически таким образом, чтобы, во-первых, каждый разметчик оценивал каждое предложение не более одного раза, а во-вторых, каждое предложение или фразеологизм оказались бы в комплектах хотя бы трёх разметчиков. В комплектах для разметки фразеологизмов для удобства разметчиков были также приведены словарные значения фразеологизмов.

Для разметчиков был проведён инструктаж, во время которого их ознакомили с предметной областью и руководствами по разметке.

В руководстве для задачи разметки предложений по тональности было указано, что тональность предложения показывает, как автор относится к теме предложения. Были выделены четыре класса тональности предложений и класс предложений с неопределимой (сомнительной) тональностью и приведены их отличительные черты, а также примеры предложений каждого из классов. Далее приведена выдержка из руководства в части описания различных классов.

- Положительные предложения, в которых автор выносит положительные оценки или выражает положительные эмоции. В положительном предложении могут упоминаться отрицательные оценки или эмоции, но автор должен явно указать, что их не разделяет. Примеры положительных предложений:

[2] https://opencorpora.org
[3] https://ru.wiktionary.org

Листьев достиг той раскованности в кадре, о какой раньше никто не помышлял.

На его взгляд, руководство, организовавшее компанию по достижению цели, заслуживает самой высокой похвалы.

Мне очень понравился фильм, а тот, кто говорит, что он скучный, просто ничего не понял.

- Отрицательные предложения, в которых автор выносит отрицательные оценки или выражает отрицательные эмоции. Положительно характеризующая тему информация отсутствует, либо является несущественной. Примеры отрицательных предложений:

  В целом получилось так, что проиграли все.

  Выяснилось, что отверстие водостока в шикарной душевой кабинке забито и поэтому принять душ было совершенно невозможно.

- Предложения со смешанной тональностью, в которых автором выносятся как положительные, так и отрицательные оценки и выражаются эмоции, соответствующие авторской позиции. Данный класс был выделен специально для предложений, являющихся тональными (выражающих авторские эмоции и содержащих авторские оценки), при этом не относимых однозначно к положительным или отрицательным. Примеры таких предложений:

  Героев «Чайки», слабых и не слишком замечательных, но всё-таки порядочных людей, Акунин изобразил порочными и преступными.

  Сделка между Google и RM получила летом хорошее освещение в медиа, но не стала сенсацией.

  Виктор Пелевин пишет не в пример хуже Акунина, но у Пелевина есть что сказать читателю.

- Нейтральные предложения, в которых автор не выносит своих оценок и не выражает согласия с чужими. Нейтральные предложения безэмоциональны, в них просто «сухо» сообщается о некоторых фактах, авторское же отношение к теме не проявляется. Примеры нейтральных предложений:

  Информация о том, что фильм провалился в прокате, не подтвердилась.

  По традиции страна-хозяйка мирового первенства освобождается от отборочного турнира.

  Газета «Пражский экспресс» утверждает, что актёры вели себя корректно, спектакль срывать не собирались, а лишь высказали требование о выплате гонораров.

- Предложения с неопределимой (сомнительной) тональностью, мысль автора в которых неясна, например, из-за того, что предложение неполное и без контекста не имеет смысла. Примеры таких предложений:

  Автор вопроса уверен. . . , или история повторяется.

  Ведь русские могли отходить до Урала.

  Потому что это чтение с самого начала слишком музыка.

Также руководство содержало ряд общих рекомендаций:

- считывать положительное либо отрицательное отношение автора вне контекста и возникающих ассоциаций;
- относить к классу смешанной тональности предложения, в которых авторское отношение однозначно есть, но неясно, положительное оно или отрицательное;

- относить к нейтральным целостные предложения, содержащие законченную мысль, но не выражающие авторское отношение; если же мысль не закончена или предложение не целостно, то считать тональность предложения неопределимой;
- при сомнениях ориентироваться на эмоциональную составляющую предложений;
- при возникновении вопросов не обсуждать их с другими разметчиками, чтобы не влиять на их восприятие.

Руководство для задачи разметки фразеологизмов по тональности было составлено аналогично; основное отличие состояло в том, что были оставлены только три класса тональности — фразеологизмы с положительной тональностью, фразеологизмы с отрицательной тональностью и нейтральные фразеологизмы. Класс смешанной тональности не использовался из-за того, что фразеологизмы гораздо короче предложений, и в них не может проявляться одновременно как положительное, так и отрицательное отношение. По схожей причине не использовался и класс неопределимой тональности — поскольку фразеологизмы достаточно коротки, а авторское отношение в них проявляется достаточно явно, все случаи, когда непонятно, положительное отношение проявляет автор или отрицательное, относятся к нейтральному классу.

Руководство для задачи разметки предложений по наличию иронии содержало примеры предложений, содержащих иронию. Также перед началом выполнения задачи разметчикам требовалось изучить определение иронии из словаря лингвистических терминов под редакцией Т. В. Жеребило [13].

Характеристики полученных данных приведены в таблице 1. Значительный размах числа оценок каждого разметчика в задачах разметки фразеологизмов по тональности и предложений по иронии обусловлен тем, что некоторые разметчики не выполнили предложенные им задания до конца.

Table 1. Corpora and annotation characteristics

Таблица 1. Характеристики корпусов и их разметки

Тип разметки	Объём корпуса	Число разметчиков	Число классов	Число оценок каждого элемента	Число оценок каждого разметчика
предлож./тональность	11608	18	5	3–5	2214–2218
фразеолог./тональность	1253	12	3	3–5	140–354
предлож./ирония	35013	14	2	1–3	1922–4000

3. Методика статистической обработки данных

Для исследования выставленных разметчиками оценок использовалась следующая методика.

1. Изучалось распределение оценок по классам у разных разметчиков. Поскольку предложения и фразеологизмы распределялись случайно, а объём индивидуальных комплектов для разметки был достаточно велик, можно ожидать, что при одинаковом понимании всеми разметчиками задачи доли оценок каждого класса для всех разметчиков будут приблизительно одинаковы. Следовательно, если доли оценок какого-либо класса у разметчиков значительно отличаются, это даёт основания считать, что разметчики решали задачу по-разному. Исследование распределения оценок по классам проводилось графическим методом на основе построенных для каждого класса гистограмм распределения доли оценок этого класса и диаграмм размаха. Также с помощью критерия хи-квадрат было проверено, совпадает ли распределение оценок по классам у каждого из разметчиков с распределением по классам всех оценок всех разметчиков. Несовпадение распределений также даёт основания считать, что способ выполнения задания у разметчика значительно отличается от способов остальных разметчиков.

2. Оценивалось согласие оценок, выставленных разными разметчиками. Если мнения разных разметчиков относительно класса одного предложения или фразеологизма часто расходятся, это может свидетельствовать как о различном понимании разметчиками задачи, так и о затруднениях в единообразном следовании инструкциям, например, из-за слишком высокой роли индивидуальных особенностей восприятия в процессе определения тональности. Согласие оценивалось на основе двух показателей: доли предложений, для которых совпали мнения всех оценивавших их разметчиков, и альфы Криппендорфа. Альфа Криппендорфа показывает, насколько согласие между разметчиками близко к полному, и рассчитывается по формуле
$$\alpha = \frac{A_o - A_e}{1 - A_e},$$
где A_o — количество совпавших оценок разметчиков; A_e — количество оценок, которые совпали бы, если бы разметчики выставляли оценки случайно [14]. В работе [15], выделяют следующие эмпирические границы для различных степеней согласия:
   - $\alpha \leq 0.2$ — согласие практически отсутствует;
   - $0.2 < \alpha \leq 0.4$ — слабое согласие;
   - $0.4 < \alpha \leq 0.6$ — умеренное согласие;
   - $0.6 < \alpha \leq 0.8$ — существенное согласие;
   - $\alpha > 0.8$ — практически полное согласие.

   Нужно отметить, что альфа Криппендорфа учитывает различия в оценках нескольких разметчиков, а не только их полное совпадение. Например, она будет выше в случае, когда среди трёх разметчиков оценки только двух совпали, чем в случае, когда все три разметчика дали разные оценки.

3. Для разметки предложений по тональности оценивалось, насколько сильно улучшится согласие, если отбросить предложения, тональность которых была отмечена как неопределимая хотя бы одним разметчиком, и предложения, тональность которых хотя бы один разметчик оценил как положительную, а другой — как отрицательную. Значительное (более, чем на 0.05) улучшение показателей согласия даст основания считать, что существенная часть несовпадающих оценок разметчиков относится именно к таким предложениям. Поскольку в речи не должно быть слишком много как предложений, тональность которых определить невозможно, так и предложений, которые одному человеку кажутся положительными, а другому отрицательными, если в разметке таких случаев оказывается много, то, возможно, руководства по разметке содержат некорректное или плохо сформулированное понятие тональности.

4. Результаты статистической обработки

4.1. Результаты обработки разметки предложений по тональности

По гистограммам распределения долей оценок каждого класса оценок и диаграмме размаха (рис. 1) явных выбросов, которые бы свидетельствовали о том, что один из разметчиков решал задачу определения предложений некоторого класса существенно иначе, чем остальные, не наблюдается. У всех разметчиков доли оценок положительного и отрицательного классов и класса смешанной тональности достаточно близки, с медианами 0.15 для положительной, 0.30 для отрицательной и 0.10 для смешанной тональности. Доля нейтральных оценок варьируется достаточно сильно — от 0.30 до 0.80. Доля предложений, тональность которых определить не получилось, изменяется ещё сильнее: у половины разметчиков она не превышает 0.01, однако у другой половины практически равномерно распределена на промежутке от 0.01 до 0.07, т. е. некоторые разметчики значительно чаще считали тональность предложения неопределимой, чем остальные. По-видимому, эти разметчики чувствовали неуверенность при решении задачи.

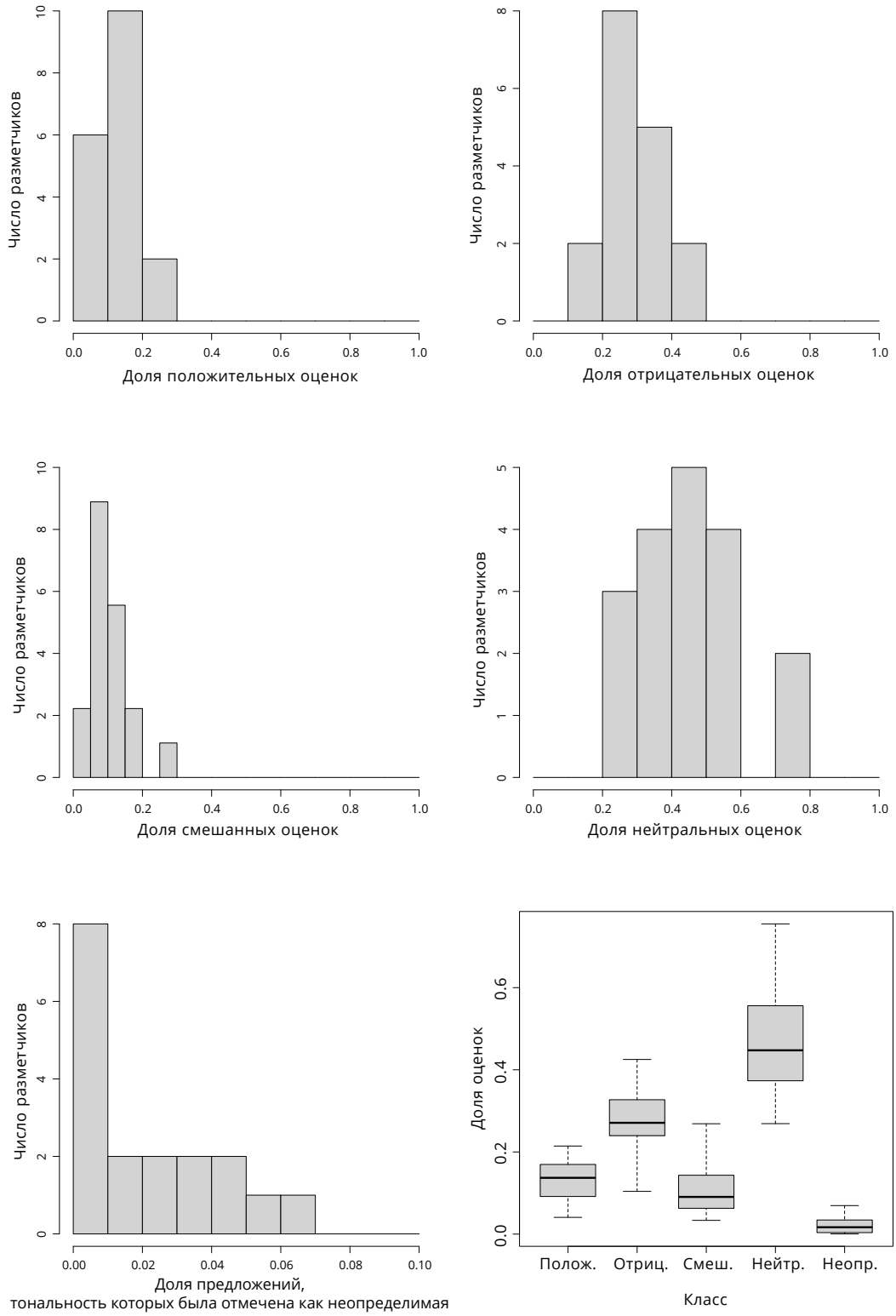

Fig. 1. Distribution of marks classes in the sentences sentiment markup

Рис. 1. Распределение классов оценок в разметке предложений по тональности

Для каждого из разметчиков критерий хи-квадрат показывает наличие значимых различий между распределениями по классам оценок этого разметчика и оценок всех остальных разметчиков. Следовательно, есть основания считать, что способы выполнения задания у разных разметчиков существенно отличались друг от друга, в первую очередь — из-за разного понимания нейтральной тональности.

Доля предложений, для которых совпали мнения всех оценивавших их разметчиков, составила 0.40, что является признаком достаточно слабого согласия в оценках разметчиков. Альфа Криппендорфа составила 0.35, что также соответствует слабому согласию.

Предложений, тональность которых хотя бы один разметчик отметил как неопределимую, оказалось 831 (7 % всех предложений). После их удаления доля предложений, для которых совпали мнения всех оценивавших их разметчиков, увеличилась на 0.03, до 0.43; альфа Криппендорфа также увеличилась на 0.03, до 0.38. Кроме того, в корпусе оказалось 338 предложений, отмеченных хотя бы одним разметчиком как имеющие положительную тональность, и хотя бы одним — как отрицательную (3 % всех предложений). После их удаления осталось 10454 предложения, т. к. тональность некоторых предложений разными разметчиками была отмечена и как неопределимая, и как положительная, и как отрицательная. Доля предложений, для которых совпали мнения всех оценивавших их разметчиков, увеличилась на 0.01, до 0.44, а альфа Криппендорфа — на 0.02, до 0.40. Общее увеличение доли предложений, для которых совпали мнения всех разметчиков, составило 0.04, а альфы Криппендорфа — 0.05. В итоге показатели согласия повысились на 0.04–0.05, и согласие осталось достаточно слабым.

4.2. Результаты обработки разметки фразеологизмов по тональности

На гистограммах распределения долей оценок каждого класса и диаграмме размаха (рис. 2) явных выбросов не наблюдается. Доля положительных оценок для всех разметчиков приблизительно одинакова, в среднем составляет 0.20. Доли отрицательных и нейтральных оценок изменяются сильнее — от 0.30 до 0.70 и от 0.10 до 0.60 соответственно. Особенно стоит отметить распределение доли нейтральных оценок: при медиане 0.35 трое из двенадцати разметчиков посчитали нейтральными менее 0.2 оцениваемых фразеологизмов. Такая аномалия свидетельствует о том, что некоторые разметчики, по-видимому, определяли фразеологизмы нейтральной тональности существенно иначе, чем остальные.

Проверка совпадения распределений оценок по классам с помощью критерия хи-квадрат показала, что для 4 из 12 разметчиков нет значимых различий между распределением оценок данного разметчика и оценок всех остальных разметчиков, а для 8 из 12 разметчиков есть.

Доля фразеологизмов, для которых совпали мнения всех оценивавших их разметчиков, составила 0.60; альфа Криппендорфа — 0.54, что соответствует умеренному согласию.

Следовательно, задача разметки фразеологизмов по тональности была понята и выполнялась разметчиками гораздо более единообразно, чем задача разметки предложений по тональности. Об этом свидетельствуют и более единообразные распределения оценок разметчиков по классам, и значительно более высокие показатели согласия. В то же время согласие между разметчиками далеко от полного. Можно было ожидать, что тональность фразеологизмов будет выражаться достаточно ярко и более определённо по сравнению с полными предложениями, однако относительно большого их числа разметчики всё же не сошлись во мнениях.

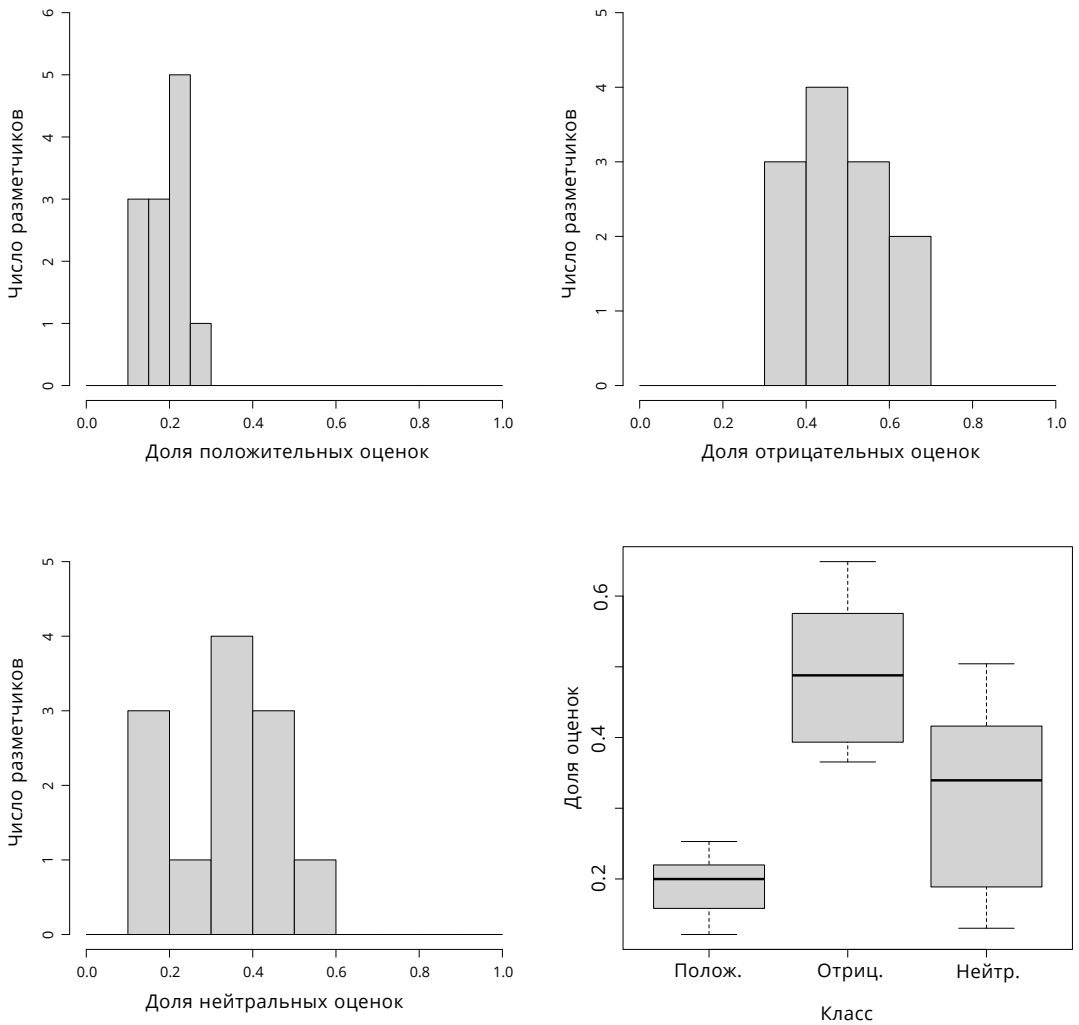

Fig. 2. Distribution of marks classes in the phrasemes sentiment markup

Рис. 2. Распределение классов оценок в разметке фразеологизмов по тональности

4.3. Результаты разметки предложений по наличию иронии

На гистограмме распределения доли предложений, отмеченных как содержащие иронию (рис. 3), явно видны три разметчика, которые отмечали иронию в экстремально высоком числе предложений (0.26, 0.33 и 0.50). Экспертная валидация подтвердила, что их разметка не соответствует инструкциям.

Чтобы проверить, насколько значительной проблемой может быть присутствие в корпусе такой некорректной разметки, были рассчитаны показатели согласия для оценок всех разметчиков, а также всех разметчиков, за исключением тех, кто не следовал инструкциям. Во втором случае доля предложений с полным совпадением оценок увеличилась на 0.06 и достигла 0.99. Учитывая, что альфа Криппендорфа в данном случае не показательна из-за слишком сильного дисбаланса классов [16], можно говорить о том, что удаление оценок разметчиков, не следовавших предложенным инструкциям, значительно улучшило согласие.

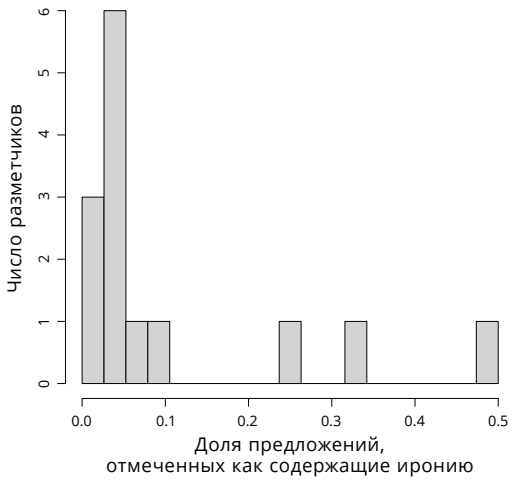

Fig. 3. Distribution of share of irony marks

Рис. 3. Распределение доли предложений, отмеченных как содержащие иронию

5. Обсуждение результатов

Обработка результатов разметки показала, что согласие между разметчиками сильно зависит от сложности задачи. Для наиболее сложной задачи разметки предложений по тональности согласие оказалось достаточно слабым, а для более простой задачи разметки фразеологизмов по тональности — значительно более сильным. Наилучшим же оказалось согласие в задаче разметки предложений по наличию иронии, которая была самой простой из выполненных разметчиками.

Одной из причин несогласия между разметчиками может быть несовершенство инструкций по разметке. Например, тональность предложения «А ежели кому кажется, что я ханжа и зануда, то так оно и есть» по формальным признакам, приведённым в инструкции, может быть определена как отрицательная, так как есть согласие автора с отрицательной оценкой и явно считываются отрицательные эмоции, хотя правильнее будет отметить её как неопределимую, поскольку в предложении присутствует ирония. Встретив такое предложение в тексте, часть разметчиков будет ориентироваться на формальные инструкции, а часть — на здравый смысл, что в итоге приведёт к несогласию. Следовательно, можно попробовать улучшить согласие, точнее сформулировав решаемую задачу и улучшив инструкции. Результаты анализа разметки показывают, что для более простой задачи определения иронии уже оказалось возможным выработать качественные инструкции, и несогласие между разметчиками было обусловлено, по-видимому, индивидуальными особенностями восприятия. В задаче же определения тональности предложений оказалось, что наиболее сильные расхождения между разметчиками касаются нейтрального класса тональности, который необходимо изучить подробнее для того, чтобы сформулировать более точные инструкции. Также следует изучить причины достаточно сильной вариации доли предложений, тональность которых определить не удалось.

Существенной проблемой является невозможность даже в случае явно некорректных оценок разметчиков определить, были ли они вызваны несовершенством инструкций, недостаточно точной постановкой задачи или же невнимательностью или недобросовестностью самого разметчика. Например, один из разметчиков оценил тональность предложения «В книге Аствацатурова определённой темы нет, как нет и сюжета, нет, кажется, и смысла» как нейтральную, несмотря на то, что в нём прослеживается отрицательное авторское отношение к теме — книге Аствацатурова. Разметчик мог допустить такую ошибку как из-за невнимательности или пренебрежения инструкциями, так

и из-за того, что инструкции рекомендовали ориентироваться на эмоциональную составляющую предложений, а предложение не содержит яркого проявления отрицательных эмоций. Организаторам процесса разметки желательно понимать причины ошибок разметчиков, чтобы в зависимости от этого либо уточнять формулировку задачи и инструкции, либо, если кто-то из разметчиков слишком часто выносит оценки не в соответствии с инструкциями, не учитывать оценки этого разметчика. Кроме того, для построения в будущем корпусов с разметкой по тональности была бы крайне полезной возможность определять, определение тональности каких именно предложений вызывает у разметчиков наибольшие сложности, чтобы на основе этой информации целенаправленно дорабатывать инструкции. Также есть необходимость в формализованной и, желательно, не требующей большого количества ресурсов методике поиска разметчиков, не следующих инструкциям. Как показала обработка разметки предложений по наличию иронии, разметка, не соответствующая инструкциям, может достаточно сильно снижать согласие.

Можно предположить, что корпус, сформированный на основе оценок нескольких разметчиков, лучше, чем сформированный на основе оценок одного разметчика, подходит для решения практических задач, поскольку в нём «сглаживаются» особенности индивидуального восприятия. Чтобы проверить, влияет ли количество разметчиков на качество работы алгоритмов анализа тональности, был проведён следующий эксперимент. Были взяты два корпуса — 9697 предложений с разметкой каждого предложения только одним человеком (первым разметчиком, оценивавшим это предложение) и 4488 предложений для которых сошлись оценки 3–5 разметчиков. На обоих корпусах была проведена оценка качества работы рекурсивного алгоритма анализа тональности [17] и RuBERT [18]. Для рекурсивного алгоритма F_1-мера на корпусе с разметкой только одним человеком составила 0.57; на корпусе предложений, для которых сошлись мнения всех разметчиков, — 0.69. Для RuBERT F_1-мера на корпусе с разметкой только одним человеком составила 0.59, на корпусе предложений, для которых совпали мнения всех разметчиков — 0.72. Следовательно, число разметчиков и согласие между ними могут оказывать существенное влияние на применение корпуса для оценки качества работы алгоритмов.

Предложим рекомендации по использованию результатов разметки предложений по тональности, полученных с использованием описанной выше методики разметки. В первую очередь необходимо отбрасывать предложения, тональность которых не смог определить хотя бы один разметчик, поскольку, скорее всего, они либо неполные и не содержат законченной авторской мысли, либо их тональность невозможно было определить на основе предложенных инструкций, и разметчики, которые отметили тональность предложения как положительную, отрицательную, нейтральную или смешанную, сделали это не на основе инструкций — в любом случае для практического использования оценки тональности таких предложений бесполезны. Также стоит отбросить предложения, тональность которых хотя бы один разметчик отметил как положительную, и хотя бы один — как отрицательную, потому что если для предложения были вынесены противоположные оценки, то, скорее всего, для определения тональности этого предложения инструкции оказались бесполезными, а значит, такое предложение лучше отбросить. Если после этого согласие разметчиков окажется практически полным, т. е. доля предложений, для которых совпали оценки всех разметчиков, и альфа Криппендорфа будут выше 0.80, то в качестве тональности каждого предложения можно рассматривать оценку, данную подавляющим большинством разметчиков: 2 из 3, 3 из 4, 3 и более из 5. Если же согласие окажется хуже, то следует использовать только те предложения, для которых совпали мнения всех оценивавших их разметчиков. Нужно отметить, что при росте числа разметчиков эти рекомендации следует пересмотреть, поскольку, когда каждое предложение оценивает, например, 10 человек, вероятность того, что хотя бы один из них не сможет определить его тональность, становится достаточно высокой.

Также представляет интерес изучение вопроса о том, что такое качество разметки и что делает разметку качественной. Определить это важно, поскольку без этого невозможно понять, к какой разметке следует стремиться. В настоящий момент известна только одна черта качественной разметки — она единообразна, т. е. выполнена в соответствии с единым набором инструкций, и, как следствие, сильным согласием [19]. Остальные же свойства, или их отсутствие, требуют изучения и формулирования.

В задаче разметки предложений по тональности разметчики выставили значительно больше нейтральных оценок, чем положительных, отрицательных и смешанных. Скорее всего, причина такого дисбаланса — большое количество новостных и информационно-аналитических текстов, имеющих нейтральную тональность. Также разметчики выставили гораздо больше отрицательных оценок, чем положительных и смешанных. Можно предположить, что это обусловлено достаточно большим количеством вошедших в корпус записей из блогов и комментариев, поскольку их гораздо чаще пишут, когда автора что-то не устраивает, чем когда всё устраивает. Этот вопрос, так же, как и соотношение фразеологизмов с положительной и отрицательной тональностью, требует отдельного исследования.

Заключение

В данной работе рассмотрена организация разметки волонтёрами трёх корпусов — корпуса предложений, размеченных по общей тональности на 4 класса, корпуса фразеологизмов, размеченных по тональности на 3 класса, и корпуса предложений, размеченных по наличию или отсутствию иронии. На основе имеющихся знаний о предметной области для каждой из задач были составлены инструкции для разметчиков. Также была выработана методика статистической обработки результатов разметки для оценки согласия между разметчиками. Для разметки предложений по наличию иронии показатели согласия оказались достаточно высокими, по-видимому, за счёт достаточно качественных инструкций и лёгкости самой задачи. При разметке предложений по общей тональности согласие оказалось достаточно слабым по причине сложности задачи, в частности, ввиду затруднений при выделении отличительных признаков нейтральных предложений. В то же время для 4488 предложений совпали мнения всех оценивавших их разметчиков, и такой объём корпуса представляется достаточным для решения большинства прикладных задач. При разметке фразеологизмов по тональности согласие оказалось значительно выше, чем при разметке предложений по тональности, что показывает, что предложенные инструкции для разметчиков достаточно хороши для более простых задач.

Авторы выражают благодарность д. э. н. Елене Михайловне Спиридоновой за содержательные замечания по тексту работы.

References

[1] V. Masoumi, M. Salehi, H. Veisi, G. Haddadian, V. Ranjbar, and M. Sahebdel, *Telecrowd: A crowdsourcing approach to create informal to formal text corpora*, 2020. arXiv: 2004.11771 [cs.SI].

[2] E. Mitiagina, M. Borodataya, E. Volchenkova, N. Ershova, M. Luchinina, and E. Kotelnikov, "Russian Text Corpus of Intimate Partner Violence: Annotation Through Crowdsourcing", in *7th International Conference on Electronic Governance and Open Society: Challenges in Eurasia. EGOSE 2020*, Springer, 2020, pp. 306–321.

[3] S. Mohammad, "A practical guide to sentiment annotation: Challenges and solutions", in *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*, Association for Computational Linguistics, 2016, pp. 174–179.

[4] S. M. Mohammad, P. Sobhani, and S. Kiritchenko, "Stance and Sentiment in Tweets", *Special Section of the ACM Transactions on Internet Technology on Argumentation in Social Media*, vol. 17, no. 3, pp. 1–23, 2017.

[5] B. R. Chakravarthi, V. Muralidaran, R. Priyadharshini, and J. P. McCrae, "Corpus Creation for Sentiment Analysis in Code-Mixed Tamil-English Text", in *Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020)*, 2020, pp. 202–210.

[6] K. Krippendorff, *Content analysis: an introduction to its methodology*. Thousand Oaks, CA: SAGE Publications, Inc., 2013.

[7] Y. Zhao, B. Qin, and T. Liu, "Creating a fine-grained corpus for chinese sentiment analysis", *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 36–43, 2014.

[8] J. Cohen, "A coefficient of agreement for nominal scales", *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[9] J. Bu, L. Ren, S. Zheng, Y. Yang, J. Wang, F. Zhang, and W. Wu, "ASAP: A Chinese Review Dataset Towards Aspect Category Sentiment Analysis and Rating Prediction", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, 2021, pp. 2069–2079.

[10] M. Navas-Loro, V. Rodríguez-Doncel, I. Santana-Perez, and A. Sánchez, "Spanish Corpus for Sentiment Analysis Towards Brands", in *Speech and Computer. SPECOM 2017*, A. Karpov, R. Potapova, and I. Mporas, Eds., Springer International Publishing, 2017, pp. 680–689.

[11] J. L. Fleiss, "Measuring nominal scale agreement among many raters", *Psychological bulletin*, vol. 76, no. 5, p. 378, 1971.

[12] A. Rogers, A. Romanov, A. Rumshisky, S. Volkova, M. Gronas, and A. Gribov, "RuSentiment: An enriched sentiment analysis dataset for social media in Russian", in *Proceedings of the 27th international conference on computational linguistics*, 2018, pp. 755–763.

[13] T. V. Zherebilo, *Slovar lingvisticheskih terminov*. Nazran: OOO Piligrim, 2010, in Russian.

[14] K. Krippendorff. (2008). Computing Krippendorff's alpha-reliability, [Online]. Available: https://repository.upenn.edu/asc_papers/43/ (visited on 01/17/2023).

[15] J. Hughes, "krippendorffsalpha: An R package for measuring agreement using Krippendorff's alpha coefficient", *The R Journal*, vol. 13, no. 1, pp. 413–425, 2021.

[16] L. A. Jeni, J. F. Cohn, and F. De La Torre, "Facing imbalanced data–recommendations for the use of performance metrics", in *2013 Humaine association conference on affective computing and intelligent interaction*, IEEE, 2013, pp. 245–251.

[17] A. Y. Poletaev and I. V. Paramonov, "Recursive sentiment detection algorithm for Russian sentences", *Modelirovanie i Analiz Informatsionnykh Sistem*, vol. 29, no. 2, pp. 134–147, 2022, in Russian.

[18] S. Smetanin and M. Komarov, "Deep transfer learning baselines for sentiment analysis in Russian", *Information Processing & Management*, vol. 58, no. 3, p. 102 484, 2021.

[19] R. Artstein and M. Poesio, "Inter-coder agreement for computational linguistics", *Computational linguistics*, vol. 34, no. 4, pp. 555–596, 2008.