Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей (05.13.11)

УДК 004.048
DOI: 10.24160/1993-6982-2019-5-117-128

Автоматическая правка ошибок правописания

О.В. Бартеньев, Д.А. Титов

Рассмотрены информационное обеспечение и алгоритмы программного приложения поиска и исправления орфографических и грамматических ошибок в текстах, написанных на русском языке. Работа приложения обеспечивается базой данных, для заполнения таблиц которой используются морфологический словарь русского языка, содержащий более четырех миллионов словоформ, и тексты различных жанров.
Перед правкой текст разбивается на фрагменты. В качестве разделителей текста на фрагменты используются знаки препинания, полученные фрагменты разбираются и правятся независимо друг от друга.
Правка текста выполняется в два этапа. На первом этапе с использованием метода орфографической коррекции на основе алгоритма симметричного удаления исправляются орфографические ошибки и ошибки, вызванные неправильным словообразованием. Для каждого слова с ошибкой на первом этапе правки текста формируется список кандидатов на замену. В качестве замещающего слова выбирается кандидат с наименьшей ценой замены — показателя, характеризующего близость заменяемого слова и кандидата. При наличии нескольких равноправных замен предпочтение отдается кандидату с наибольшим числом вхождений в тексты, ранее использованных для заполнения базы данных приложения.
На втором этапе правки исправляются некоторые виды грамматических ошибок. Правка выполняется на основе прецедентов — случаев употребления пары «слово – следующее слово» в прошедших редакционную правку текстах. Используя найденные в базе данных прецеденты, приложение выделяет слова, подлежащие замене. Так же как и на первом этапе, выбор замещающего слова осуществляется из списка кандидатов, однако замены не будет, если ее цена превысит допустимое значение.
Правка текста может выполняться как автоматически, так и в режиме интерактивного выбора замещающего слова. На тестовом наборе данных, содержащем как орфографические, так и грамматические ошибки, приложение исправляет больше слов, чем Microsoft Word и Яндекс-спеллер.

Ключевые слова: база данных, автоматическая правка текста, алгоритм симметричного удаления, прецедент, орфографическая и грамматическая ошибки.

Для цитирования: Бартеньев О.В., Титов Д.А. Автоматическая правка ошибок правописания // Вестник МЭИ. 2019. № 5. С. 117—128. DOI: 10.24160/1993-6982-2019-5-117-128.

Automatic Correction of Spelling Errors

O.V. Barten′ev, D.A. Titov

Information support and algorithms of a software application for searching and correcting spelling and grammatical errors in texts written in Russian are considered. The application's operation is supported by a database the tables of which are filled using the morphological dictionary of the Russian language containing more than four million word forms and texts of different genres.
Before being subjected to correction, the text is divided into fragments; punctuation marks are used as text separators. The obtained text fragments are checked and corrected independently of each other.
The text is corrected in two stages. At the first stage, spelling errors and errors caused by incorrect word formation are corrected using the spelling correction method based on the symmetric deletion algorithm. For each word with an error, a list of candidates for replacement is formed at the first text correction stage. The candidate with the lowest replacement cost — an indicator characterizing the proximity of the

word to be replaced and the candidate — is chosen as the replacing word. If there are several candidates with the equal replacement cost, preference is given to the candidate with the highest number of entries in the texts that were previously used to fill the application database. At the second text correction stage, certain types of grammatical errors are corrected. The correction is carried out on the basis of precedents — cases of using the "word — next word" pair in the texts that have undergone editorial correction. By using the precedents found in the database, the application highlights the words to be replaced. By analogy with the first text correction stage, the replacing word is chosen from the list of candidates, but the replacement will not be done if its cost exceeds the permissible value.
The text can be corrected both automatically and by interactively selecting a replacing word. In processing a test data set containing both spelling and grammatical errors, the application corrects more words than the Microsoft Word and Yandex-speller do.

Key words: database, automatic text correction, symmetric deletion algorithm, precedent, spelling error, grammatical error.

For citation: Barten′ev O.V., Titov D.A. Automatic Correction of Spelling Errors. Bulletin of MPEI. 2019;5:117—128. (in Russian). DOI: 10.24160/1993-6982-2019-5-117-128.

Введение

Современные средства проверки правописания могут находить орфографические и некоторые разновидности грамматических ошибок, но не справляются с другими видами: пунктуационными, речевыми, стилистическими, логическими и фактическими [1 – 3]. Так, ни Microsoft Word, ни Яндекс-спеллер [4] не найдут грамматические ошибки в следующих текстах: «многие чуда техники, обоим сестрам, кто это пришла». Эти же программы посчитают, что текст «крашеный недавно забор» не содержит ошибки, в то время как первое слово текста является причастием и, следовательно, должно обладать суффиксом енн: «крашенный недавно забор». Аналогично себя ведут и другие программы, например, почтовые и поисковые, обладающие средствами проверки правописания.

Предмет работы

Рассмотрены информационное обеспечение и алгоритмы программного приложения поиска и исправления орфографических и грамматических ошибок (приложение).

Орфографическая ошибка — ошибка в слове (буквенная, постановка дефиса, слитное и раздельное написание) [3].

Грамматическая ошибка — нарушение структуры языковой единицы, вызванное нарушением норм слово- и формообразования, синтаксической связи между словами внутри предложения или словосочетания [3].

Примеры орфографических и грамматических ошибок, с которыми «справляется» приложение, приведены в табл. 1, 2, примеры неопределяемых грамматических ошибок даны в табл. 3 [3].

Таблица 1

Примеры орфографических ошибок

С ошибкой	Без ошибки
Внизапно пашел снег.	Внезапно пошел снег.
Прелетели грачи.	Прилетели грачи.
Он неуспел растроится.	Он не успел расстроиться.

Таблица 2

Примеры распознаваемых грамматических ошибок

Вид ошибки	С ошибкой	Без ошибки
Ошибочное словообразование	Трудолюбимый Равнодушество Подскользнуться Военоначальник	Трудолюбивый Равнодушие Поскользнуться Военачальник
Ошибочное образование формы прилагательного	Красивше выглядит	Красивее выглядит
Ошибочное образование формы глагола	По вечерам я дремаю у телевизора	По вечерам я дремлю у телевизора
Ошибочное образование формы существительного	Многие чуда техники	Многие чудеса техники
Ошибочное образование формы числительного	Обоим сестрам	Обеим сестрам
Ошибочное образование формы местоимения	Ихние дети	Их дети
Нарушение управления	Повествует читателей	Повествует читателям
Нарушение способа выражения сказуемого в отдельных конструкциях	Все были рады, счастливы и весёлые	Все были рады, счастливы и веселы
Нарушение согласования	Я знаком с группой ребят, увлекающимися джазом	Я знаком с группой ребят, увлекающихся джазом
Нарушение связи между подлежащим и сказуемым	Кто это пришла?	Кто это пришел?

Таблица 3

Примеры нераспознаваемых грамматических ошибок

Вид ошибки	С ошибкой	Без ошибки
Ошибки в построении предложения с причастным оборотом	Узкая дорожка была покрыта проваливающимся снегом под ногами	Узкая дорожка была покрыта проваливающимся под ногами снегом
Ошибки в построении предложения с однородными членами	Страна любила и гордилась поэтом. Построена школа и больница	Страна любила поэта и гордилась им. Построены школа и больница
Ошибки в построении предложения с деепричастным оборотом	Читая этот текст, возникает такое чувство…	Когда я читаю этот текст, возникает такое чувство…
Ошибки в построении сложного предложения	Эта книга научила меня ценить и уважать друзей, которую я прочитал еще в детстве	Эта книга, которую я прочитал еще в детстве, научила меня ценить и уважать друзей
Смешение прямой и косвенной речи	Автор сказал, что я не согласен с мнением рецензента	Автор сказал, что он не согласен с мнением рецензента
Нарушение границ предложения	Когда герой опомнился. Было уже поздно	Когда герой опомнился, было уже поздно
Нарушение видовременной соотнесённости глагольных форм	Замирает на мгновение сердце и вдруг застучит вновь	Замирает на мгновение сердце и вдруг стучит вновь
Неверное употребление местоимений	Данный текст написал В. Белов. Он относится к художественному стилю	Данный текст написал В. Белов. Текст относится к художественному стилю

База данных приложения

Работа приложения поддерживается базой данных (БД), таблицы которой предварительно заполняются в результате обработки морфологического словаря русского языка и разбора текстов, прошедших редакционную правку.

База данных приложения [5] создана на платформе SQL Server. При правке текста используются следующие таблицы:

— «Словоформы без повторов» (ТБП);
— «Укороченные словоформы» (ТУС);
— «Прецеденты» (ТП);
— «Новые слова» (ТНС);
—« Всего прецедентов» (ТВП).

Строковые поля таблиц содержат записи в нижнем регистре.

ТБП создана на основе морфологического словаря русского языка [5], содержащего 4 159 394 словоформы, путем исключения повторяющихся словоформ. Так, в морфологическом словаре слово «пила» имеет два омонима (существительное и глагол прошедшего времени), например: «Пила нашлась за печкой. Не ела не пила, а милого ждала». В ТБП слово «пила» присутствует один раз. Первоначально ТБП содержит 2 438 546 словоформ и имеет следующие поля:

— КодСлова — код слова;
— Слово.

ТУС реализована на основе ТБП: для каждой словоформы находится подмножество строк, образованных в результате удаления букв в исходном слове. Число удаляемых букв называется расстоянием редактирования. Например, для слова «корова» это подмножество с расстоянием редактирования 1 будет следующим: орова, крова, коова, корва, короа, коров. Все полученные строки добавляются в ТУС. После обработки начальной ТБП с расстоянием редактирования 1 в ТУС добавлено 27 331 576 записей.

ТУС имеет следующие поля:

— КодСлова — код слова в ТБП;
— УкороченноеСлово — укороченная словоформа.

ТП создается в результате разбора текстов, прошедших редакционную правку. Эта таблица фиксирует прецеденты вида: слово – следующее слово. В результате разбора фрагмента «мама всегда радуется гостям» в ТП будут отображены приведенные в табл. 4 прецеденты.

Таблица 4

Отражение прецедентов в ТП

Слово	Сосед	Число прецедентов
мама	всегда	1
всегда	радуется	4
радуется	гостям	2

Значение в поле «Число прецедентов» будет увеличено на 1 при каждом новом обнаружении прецедента.

В ТП входят три следующих поля:

— КодСлова — код слова в ТБП;
— КодСоседа — код слова-соседа в ТБП;
— ЧислоПрецедентов — число прецедентов.

Если при разборе текста встречено слово, отсутствующее в ТБП, то оно будет добавлено в ТБП и ТНС. Последняя таблица нужна для проверки новых слов: если слово содержит ошибку, то оно будет удалено из названных выше таблиц. Слова, прошедшие проверку, требуются для пополнения ТУС. Сразу же после пополнения ТП корректируется ТВП.

ТВП создается по ТП и имеет следующие поля:

— КодСлова — код слова в ТБП;

— ЧислоЛ — число прецедентов, зафиксированных в ТП, в которых слово с кодом КодСлова занимает первую (левую) позицию в прецеденте;

— ЧислоП — число прецедентов, зафиксированных в ТП, в которых слово с кодом КодСлова занимает вторую (правую) позицию в прецеденте.

Программы создания и заполнения таблиц БД приведены в [5].

Этапы и методы правки текста

Правка текста выполняется в два этапа. На первом исправляются орфографические и грамматические ошибки, вызванные неправильным словообразованием. Коррекция заключается в замене проверяемого слова на слово из ТБП.

Затем в исправленном тексте для каждого слова определяется значение (Истина или Ложь) атрибута Сохранить, после чего он передается на второй этап правки. На втором этапе слова с атрибутом Сохранить = Ложь проверяются на предмет наличия грамматической ошибки. При ее обнаружении приложение либо находит замену слову, либо оставляет его без изменений.

Пример. В тексте «старик пасмотрел на карову сваю» на первом этапе будут исправлены две первые ошибки: «старик посмотрел на корову сваю». На втором существительное «сваю» будет заменено на притяжательное местоимение «свою».

В результате анализа методов проверки правописания [6] для первого этапа выбран метод орфографической коррекции на базе алгоритма симметричного удаления [7].

На втором этапе грамматическая ошибка устанавливается по результатам поиска прецедентов — случаев, имевших место ранее и служащих примером или оправданием для последующих случаев подобного рода [8]. В настоящей работе прецедент — это случай употребления пары: слово – следующее слово в прошедших редакционную правку текстах. Прецеденты накапливаются в ТП в результате разбора текстов. Эта же таблица используется для поиска кандидатов на замену не имеющего прецедентов слова.

Правка текста реализуется по следующей схеме:

— выполнить первый этап правки текста;

— для каждого слова текста определить значение атрибута Сохранить;

— выполнить второй этап правки текста, проверяя слова, для которых Сохранить = Ложь;

— вывести информацию о выполненных заменах.

Замечание. Обрабатываемый текст приводится к нижнему регистру. При выводе исправленного текста прописные буквы восстанавливаются на прежнем месте.

Фрагмент текста

При разборе и правке текст разбивается на фрагменты. В качестве разделителей используются знаки препинания [9]. Затем каждый фрагмент сжимается за счет удаления словоформ, число букв в которых меньше трех. Фрагменты текста обрабатываются (разбираются и правятся) независимо. Сжатие позволяет избегать фиксации прецедентов, понижающих эффективность правки.

Пример. Если отказаться от сжатия, то при разборе второго фрагмента текста «мальчики были рады и счастливы, и весёлые они пошли домой» будет зафиксирован прецедент «и — весёлые». Тогда при правке фрагмента «мальчики были счастливы и весёлые» данным прецедентом будет оправдано присутствие слова «весёлые». Если же работать со сжатым текстом, то прецедент не будет добавлен в таблицу прецедентов, что позволит перейти к поиску замены слова. Правильная замена будет найдена, если ранее был разобран, например, фрагмент «все были счастливы веселы», полученный после сжатия текста. На самом деле, кроме прецедентов «все – были», «были – счастливы» также зафиксирован и прецедент «счастливы – веселы», благодаря которому слово «веселы» будет рассматриваться приложением как кандидат на замену слова «весёлые».

Разбивка на фрагменты позволяет устранить нежелательные прецеденты. Например, прецедент «счастливы – весёлые» зафиксирован не будет, поскольку слова счастливы и весёлые принадлежат разным фрагментам. Ошибочность фиксации прецедента подемонстрирована приведенным примером.

Каждое слово фрагмента имеет атрибут Сохранить, принимающий значение Истина, если выполнены условия сохранения слова в тексте, или Ложь — в противном случае.

Разбор фрагмента текста

Цель разбора — пополнить таблицу прецедентов. Во фрагменте могут встречаться слова, отсутствующие в ТБП, например, имена собственные, просторечия или слова с опечатками, поэтому при разборе фрагмента также пополняются ТБП и ТНС. Затем слова, добавленные в ТНС и прошедшие проверку, используются для пополнения ТУС, после чего корректируется ТВП.

Разбор фрагмента выполняется по следующему алгоритму.

```
1. Начало.
2. Расчленить фрагмент на слова и записать их в массив МассивСлов.
3. ЧислоСлов = МассивСлов.Количество() // Число слов во фрагменте текста
4. Для к = 1 По ЧислоСлов – 1 Цикл
        Слово = МассивСлов[к] // Текущее слово
        Сосед = МассивСлов[к + 1] // Правый сосед текущего слова
        КодСлова = ВЫБРАТЬ КодСлова ИЗ ТБП ГДЕ Слово = Слово
        КодСоседа = ВЫБРАТЬ КодСлова ИЗ ТБП ГДЕ Слово = Сосед
        Если КодСлова = NULL Тогда
```

```
        КодСлова = ДобавитьВТБПиВТНС(Слово)
        НовоеСлово = Истина
    КонецЕсли
    Если КодСоседа = NULL Тогда
        КодСоседа = ДобавитьВТБПиВТНС(Сосед)
        НовыйСосед = Истина
    КонецЕсли
    Если НовоеСлово ИЛИ НовыйСосед Тогда
        // Фиксируем в ТП прецедент Слово – Сосед
        ДобавитьПрецедентВТП(КодСлова, КодСоседа)
    Иначе
        // Функции ЕстьПрецедент вернет Истина, если в ТП
        // уже зафиксирован прецедент Слово – Сосед
        Если ЕстьПрецедент(КодСлова, КодСоседа) Тогда
            УвеличитьЧислоПрецедентов(КодСлова, КодСоседа)
        Иначе
            ДобавитьПрецедентВТП(КодСлова, КодСоседа)
        КонецЕсли
    КонецЕсли
КонецЦикла
```

6. Останов.

Замечание. В алгоритме для записи текста запроса употребляется псевдокод.

Функция ДобавитьВТБПиВТНС принимает в качестве параметра слово и добавляет его в ТБП и ТНС.

Функция ДобавитьПрецедентВТП принимает в качестве параметров коды слова и его соседа и фиксирует в ТП прецедент Слово – Сосед.

Функция ЕстьПрецедент принимает в качестве параметров коды слова и его соседа, выполняет запрос к ТП и возвращает Истина, если обнаружен прецедент Слово – Сосед, или Ложь — в противном случае.

Функция УвеличитьЧислоПрецедентов принимает в качестве параметров коды слова и его соседа и увеличивает в ТП на 1 число случаев Слово – Сосед, обнаруженных в текущем и ранее разобранных текстах.

После разбора всех фрагментов анализируются свежие записи ТНС и формируется список кодов слов с ошибками. Затем из ТБП, ТНС и ТП удаляются записи, соответствующие этому списку. Оставшиеся в ТНС новые слова используются для пополнения ТУС и корректировки ТВП.

Программная реализация рассмотренного и приводимых далее алгоритмов приведена в [5].

Множество кандидатов на замещение проверяемого слова

На каждом этапе правки текста слово для замещения проверяемого слова выбирается из множества слов-кандидатов (МК) на замещение. Элемент МК имеет следующие атрибуты:

— ЦЗ(слово, кандидат) — цена замены проверяемого слова на кандидат;

— ВсегоПрецедентов(кандидат) — общее число прецедентов с участием кандидата, определяемое по ТВП.

Приложение при вычислении ЦЗ учитывает следующие факторы.

ДамЛев(слово, кандидат) — расстояние Дамерау–Левенштейна между проверяемым словом и кандидатом, равное минимальному количеству операций вставки, удаления, замены и перестановки символов, превращающих одно слово в другое.

Пример. ДамЛев(сваю, свою) = 1, ДамЛев(свою, дорого) = 6.

ШтрафГласные — штраф за несовпадение числа гласных в проверяемом слове и в кандидате. Равен 0, если число гласных в проверяемом слове и в кандидате одинаково, и 1 — в противном случае.

ШтрафПрецеденты — штраф за отсутствие прецедентов кандидата в ТП. Равен 0, если обнаружены прецеденты кандидата, и 1 — в противном случае. На втором этапе правки текста данный штраф всегда равен нулю, поскольку кандидаты выбираются из прецедентов.

Таким образом,

$$\text{ЦЗ(слово, кандидат)} = \text{ДамЛев(слово, кандидат)} + \\ + \text{ШтрафГласные} + \text{ШтрафПрецеденты}.$$

При выборе замещающего слова предпочтение отдается кандидату с наименьшей ценой замены.

Первый штраф позволяет отдать приоритет кандидатам, фонетически более близким к заменяемому слову.

Пример. Заменяемое слово: програма. Кандидаты: программа и программ. По цветовым картам заменяемого слова и кандидатов (рис. 1), построенных в результате фонетического разбора слов [10], можно говорить о большей фонетической схожести заменяемого слова и кандидата программа.

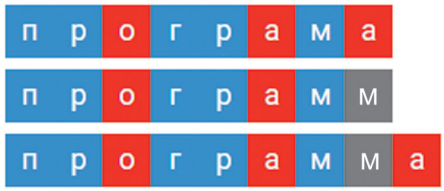


Рис. 1. Цветовые карты заменяемого слова и кандидатов

На цветовой карте голубой цвет отвечает твердым согласным звукам, красный — гласным, а серый — букве, не образующей звука в слове [11].

Второй штраф увеличивает цену замены слова на другое редко употребляемое слово. Так, словарь А С. Пушкина содержит немногим более 20 тыс. слов [12]. Активный словарь обычного человека существенно меньше. В то же время морфологический словарь русского языка включает более 4 млн слов, т. е. вероятность появления большого числа слов в текстах, предназначенных для широкого читателя, незначительна. Подобные слова можно отнести к разряду редких.

В приложении слово считается редким, если оно не имеет прецедентов. Встретив редкое слово, приложение при расчете цены замены (ЦЗ) увеличивает рассчитываемое значение на 1.

Пример. Заменяемое слово: предже. Кандидаты: преде и прежде.

ДамЛев(предже, преде) = 1;
ДамЛев(предже, прежде) = 1.

Число гласных во всех словах одинаково.

ВсегоПрецедентов(преде) = 0;
ВсегоПрецедентов(прежде) = 348.
ЦЗ(предже, преде) = 1 + 0 + 1;
ЦЗ(предже, прежде) = 1 + 0 + 0.

Для замены будет выбран кандидат прежде.

Замечание. В качестве ЦЗ можно использовать и иной показатель, например, фонетическую близость слов [6].

Атрибут ВсегоПрецедентов определяется для элемента МК в результате выполнения запроса, имеющего следующий текст:

```
ВсегоПрецедентов = ВЫБРАТЬ (ЧислоЛ + ЧислоП) ИЗ ТВП
    ГДЕ КодСлова = КодКандидата
```

Цена замены характеризует близость слов, а ВсегоПрецедентов — частоту их употребления в ранее разобранных текстах.

Виды прецедентов

Отдельно взятое слово может находиться в прецеденте либо на первой, либо на второй позициях. Прецедент, в котором взятое слово находится в первой позиции, назовем левым, а прецедент, в котором взятое слово находится во второй позиции, — правым.

Примеры левого и правого прецедентов слова дремлет: дремлет – река и тихо – дремлет.

Правила сохранения слова в тексте

На первом этапе правки текста проверяемое слово сохраняется в тексте, если оно найдено в ТБП.

На втором этапе обрабатываются фрагменты с двумя и большим числом слов. Для каждого слова определяется значение его атрибута Сохранить. Первое, последнее и срединные слова имеют разные алгоритмы определения значения данного атрибута.

Случай первого слова фрагмента:

```
// Слово – первое слово фрагмента; Слово2 – второе слово фрагмента
// Слово3 – третье слово фрагмента; ЧислоСлов – число слов в фрагменте
1. Начало.
2. Сохранить = Ложь
3. Сформировать МЛ – множество слов в левых прецедентах Слово.
4. Если Слово2 ∈ МЛ Тогда
        Сохранить = Истина
   ИначеЕсли ЧислоСлов > 2 Тогда
        Сформировать МП3 – множество слов в правых прецедентах Слово3.
        Если Слово2  ∈ МП3 Тогда
            Сохранить = Ложь
        КонецЕсли
   Иначе
        Сохранить = Истина
   КонецЕсли
5. Останов.
```

Согласно алгоритму, первое слово фрагмента сохраняется если:

— есть прецедент Слово – Слово2;

— во фрагменте два слова и нет прецедента Слово – Слово2 (отдается приоритет первому слову, будет выполнена попытка заменить Слово2);

— во фрагменте более двух слов и нет прецедентов Слово – Слово2 и Слово2 – Слово3.

В противном случае выполняется попытка найти замену первому слову фрагмента.

Случай срединного слова фрагмента:

```
// Слово – текущее срединное слово
// СловоЛ – слово, стоящее перед Слово; СловоП – слово, стоящее после Слово
1. Начало.
2. Сформировать МЛ – множество слов в левых прецедентах СловоЛ.
3. Сформировать МП – множество слов в правых прецедентах СловоП.
4. Если Слово ∈ МЛ ∩ МП Тогда
        Сохранить = Истина
   Иначе
        Сохранить = Ложь
   КонецЕсли
5. Останов.
```

Пример. Проверятся фрагмент «корову свою продам».

Слово = свою; СловоЛ = корову; СловоП = продам.

Левые прецеденты слова корову: корову – доит, корову – жалко, корову – свою, корову – тебе.

Правые прецеденты слова продам: пеньку – продам, птицу – продам, свою – продам, тебе – продам.

МЛ = { доит, жалко, свою, тебе }; МП = { пеньку, птицу, свою, тебе };

свою ∈ МЛ ∩ МП = { свою, тебе } → Сохранить = Истина.

При проверке фрагмента корову сваю продам имеем:

сваю ∉ МЛ ∩ МП = { свою, тебе } → Сохранить = Ложь.

Случай последнего слова фрагмента:

```
// Слово – последнее слово фрагмента; СловоЛ – предшествующее слово фрагмента
1. Начало.
2. Сформировать МЛ – множество слов в левых прецедентах СловоЛ.
3. Если Слово ∈ МЛ Тогда
        Сохранить = Истина
   Иначе
        Сохранить = Ложь
   КонецЕсли
4. Останов.
```

Замечание. Слово, в котором атрибут Сохранить = Ложь, оставляется в тексте, если не будут выполнены условия замены слова.

Первый этап правки текста

На первом этапе правки текста исправляются ошибки правописания, возникающие в результате:

— замены буквы слова (карова вместо корова);

— пропуска буквы в слове (осений вместо осенний);

— добавления буквы в слово (ландышь вместо ландыш);

— перестановки двух букв слова (предже вместо прежде);

— слитного написании двух слов (вобщем вместо в общем);

— пропуска дефиса (наконецто вместо наконец-то).

Замечание. Случай написания слова в виде двух слов (едино временно вместо единовременно) приложением не обрабатывается.

Каждое проверяемое слово, если не выполнены условия его сохранения в тексте, будет замещено кандидатом из МК, если МК ≠ ∅, формируемым по следующему алгоритму:

```
// Слово — замещаемое слово.
1. Начало.
2. МК = ∅ // Множество кандидатов на замещение слова с ошибкой.
3. ДлинаСлова = Слово.Длина // Число букв в слове Слово.
4. Для к = 1 По ДлинаСлова Цикл
       НовоеСлово = ВставитьПробел(к, Слово)
       Кандидат = ВЫБРАТЬ Слово ИЗ ТБП ГДЕ Слово = НовоеСлово
       Если Кандидат != NULL Тогда МК = МК ∪ Кандидат
       КонецЕсли
   КонецЦикла
5. Для к = 1 По ДлинаСлова Цикл
       НовоеСлово = ВставитьДефис(к, Слово)
       Кандидат = ВЫБРАТЬ Слово ИЗ ТБП ГДЕ Слово = НовоеСлово
       Если Кандидат != NULL Тогда МК = МК ∪ Кандидат
       КонецЕсли
   КонецЦикла
6. РезультатЗапроса = ВЫБРАТЬ Слово ИЗ ТУС ГДЕ УкороченноеСлово = Слово
7. Если РезультатЗапроса != NULL Тогда МК = МК ∪ РезультатЗапроса КонецЕсли
8. Для к = 1 По ДлинаСлова Цикл
   8.1. КороткоеСлово = СловоБезБуквы(к, Слово)
   8.2. РезультатЗапроса = ВЫБРАТЬ Слово ИЗ ТБП
            ГДЕ Слово = КороткоеСлово
   8.3. Если РезультатЗапроса != NULL Тогда
            МК = МК ∪ РезультатЗапроса
        КонецЕсли
   8.4. РезультатЗапроса = ВЫБРАТЬ Слово ИЗ ТУС
            ГДЕ УкороченноеСлово = КороткоеСлово
   8.5. Если РезультатЗапроса != NULL Тогда
                МК = МК ∪ РезультатЗапроса
        КонецЕсли
   КонецЦикла
9. Останов.
```

Функции ВставитьПробел и ВставитьДефис вставляют соответственно пробел или дефис после к-го символа строки Слово, а функция СловоБезБуквы удаляет к-ый символ строки Слово.

Шаг 4 алгоритма позволяет исправлять случаи слитного написания двух слов. Например, строка «вобщем» на первой итерации цикла 4 преобразовывается в строку «в общем», которая будет найдена в ТБП и добавлена в МК.

Шаг 5 позволяет исправлять случаи пропуска дефиса. Например, строка «вопервых» на второй итерации цикла 5 будет преобразована в строку «во-первых», найденную в ТБП и добавленную в МК.

Шаги 6 и 7 позволяют исправлять слова, в которых пропущена буква.

Пример. При добавлении укороченных слов, образованных от слова «речь, в ТУС попадут строки «рчь», «реь» и «реч». При правке текста «торопливая реч ручейка» строка «реч» будет найдена в ТУС, и родительское слово добавится в МК.

Шаги 8.2 и 8.3 позволяют исправлять слова, в которых лишняя буква.

Пример. При правке слова «ландышь» в цикле 8 будет получено в том числе и слово «ландыш», оно будет найдено в ТБП идобавлено в МК.

Шаги 8.4 и 8.5 позволяют исправлять слова с неверной буквой.

Пример. При добавлении в ТУС укороченных слов, образованных от слова «проект», в ТУС попадет в том числе и строка «прокт. При правке слова «проэкт» в цикле 8 будет получена строка «прокт». Она найдется в ТУС, и родительское слово «проект» поступит в МК.

Выбор замещающего слова делается из МК по следующим показателям:

— цена замены проверяемого слова на кандидат;

— общее число прецедентов (ВсегоПрецедентов) с участием кандидата.

Для замены берется слово МК с минимальным значением ЦЗ и максимальным значением ВсегоПрецедентов.

Пример. Заменяемое слово: «карова». Кандидаты на замещения этого слова показаны в табл. 5, в ней же приведены ЦЗ слова «карова» на кандидат и общее число прецедентов кандидата.

Слово «карова» будет заменено на корова, поскольку среди кандидатов с ЦЗ = 1 это слово имеет наибольшее значение ВсегоПрецедентов.

Второй этап правки текста

На втором этапе правки текста обрабатываются слова с атрибутом Сохранить = Ложь. Для каждого слова формируется МК, из которого, если МК ≠ ∅, и выбирается замещающее слово. Первое, последнее и срединные слова фрагмента имеют разные алгоритмы формирования МК.

Таблица 5

Кандидаты на замещение слова «карова»

Короткое слово	Кандидат на замещение	Цена замены	Всего прецедентов
крова	крова	2	6
""	кровка	3	0
""	кровав	3	0
""	кровам	3	0
""	кровах	3	0
""	корова	1	85
каова	какова	1	16
каров	каров	3	0
""	каюров	3	0
""	кафров	3	0
""	картов	3	0
""	карпов	2	1
""	кадров	2	1
""	икаров	3	0

Алгоритм формирования МК на замещение первого слова фрагмента:

// Слово — первое слово фрагмента; СловоП — слово, стоящее после Слово.
1. Начало.
2. Сформировать МП — множество слов в правых прецедентах СловоП.
3. МК = МП.
4. Останов.

Алгоритм формирования МК на замещение срединного слова фрагмента:

// Слово — срединное слово.
// СловоЛ — слово, стоящее перед Слово; СловоП — слово, стоящее после Слово.
1. Начало.
2. Сформировать МЛ — множество слов в левых прецедентах СловоЛ.
4. Сформировать МП — множество слов в правых прецедентах СловоП.
6. МК = МЛ ∩ МП (рис. 2).
7. Останов.

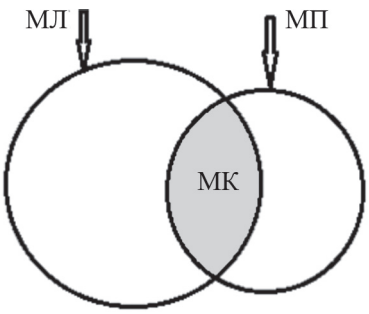


Рис. 2. МК-множество кандидатов на замещение срединного слова

Алгоритм формирования МК на замещение последнего слова фрагмента:

// Слово — последнее слово фрагмента; СловоЛ — слово, стоящее перед Слово.
1. Начало.
2. Сформировать МЛ — множество слов в левых прецедентах СловоЛ.
3. МК = МЛ.
4. Останов.

Замечание. При формировании МК для его элементов вычисляются атрибуты ЦЗ и ВсегоПрецедентов.

Выбор замещающего слова на втором этапе правки

Слово, замещающее проверяемое слово, выбирается из МК по двум показателям ЦЗ и ВсегоПрецедентов с учетом ограничения

$$ЦЗ < ЦЗМакс,$$

где ЦЗМакс — максимально допустимая цена замены проверяемого слова. Если это условие нарушено, то проверяемое слово остается в тексте.

Ограничение введено для уменьшения риска некорректных замен. Величина ЦЗМакс зависит от длины проверяемого слова (ДлинаСлова) и вычисляется по следующей формуле:

$$ЦЗМакс(Слово) = \\ =Мин(Макс(ЦенаМин, ДлинаСлова — 2), ЦенаМакс),$$

где ЦенаМин и ЦенаМакс — нижняя и верхняя границы ЦЗМакс.

Зависимость ЦЗМакс от длины слова продемонстрирована на рис. 3.

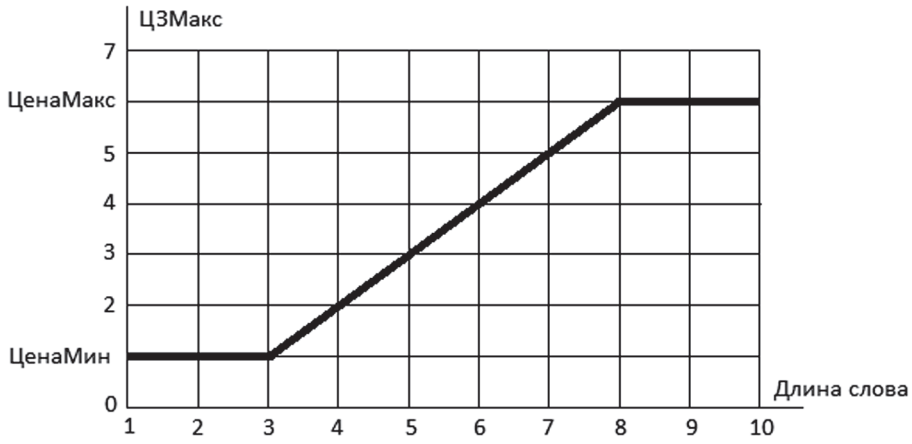


Рис. 3. Зависимость ЦЗМакс от длины слова

Замечание. В Приложении ЦенаМин = 0, ЦенаМакс = 4; значения подобраны экспериментально.

Выбор кандидата, замещающего текущее слово, и замена слова выполняются по следующему алгоритму:

// Слово — текущее слово.
// МК — множество кандидатов на замещение текущего слова (МК ≠ ∅).
// Для каждого кандидата известны ЦЗ(Слово, Кандидат) и ВсегоПрецедентов.
1. Начало.
2. Упорядочить МК по возрастанию ЦЗ и убыванию ВсегоПрецедентов.
3. Кандидат = МК[1] // Первый элемент МК
4. Если ЦЗ(Слово, Кандидат) < ЦЗМакс(Слово) Тогда
    Заменить Слово на Кандидат.
  Иначе
    Оставить Слово в фрагменте.
  КонецЕсли
5. Останов.

Пример 1. МК = {свою, тебе}; замещается слово «сваю» в фрагменте «корову сваю продам».

ЦЗ(сваю, свою) = 1 < ЦЗ(сваю, тебе) = 4, поэтому после упорядочивания МК = {свою, тебе}.

ЦЗМакс(сваю) = 4 — 2 = 2; ЦЗ(сваю, свою) < < ЦЗМакс(сваю), поэтому слово «сваю» замещается кандидатом «свою».

Показатель ВсегоПрецедентов к выбору замещающего слова привлекать не требуется.

Пример 2. МК = {дорого, тебе}; замещается слово «свою» в фрагменте «корову сваю продам».

ЦЗ(свою, тебе) = 4 < ЦЗ(свою, дорого) = 5), поэтому после упорядочивания МК = {тебе, дорого}.

ЦЗМакс(свою) = 4 — 2 = 2; ЦЗ(свою, тебе) > > ЦЗМакс(свою), поэтому «свою» не будет замещено кандидатом тебе.

Результаты тестирования

При тестировании на вход приложения подавались тексты, содержащие ошибки (тестовый набор описан в [5]). В набор вошли все приведенные в табл. 1, 2 примеры. Часть текстов с ошибками взята из [1, 2]. Прочие примеры создавались путем внесения в текст ошибок в результате замены, перестановки, добавления и удаления букв в словах. Проверялись случаи слитного написания слов и пропуска дефиса. Для сравнения все тексты проходили проверку Microsoft Word и Яндекс-спеллер.

В текстах содержались 73 орфографических и 25 грамматических ошибок. Результаты проверки приведены в табл. 6.

Замечание. При тестировании использованы актуальные на дату тестирования версии Microsoft Word и Яндекс-спеллер. Они предлагают выбрать замену из списка кандидатов (рис. 4).

Выбор первого кандидата в Microsoft Word или в Яндекс-спеллер аналогичен автоматической правке в приложении. Часть ошибок при работе устранялась в результате выбора второго или последующего кандидатов. Ошибка оставалась неисправленной, если программа либо не предъявляла правильной замены, либо не замечала ошибки.

Рассматриваемое приложение автоматически правит текст, выбирая первое слово из списка кандидатов (рис. 5).

По результатам тестирования можно сказать, что на первом этапе приложение исправило в словах ошибки, обусловленные заменами одной буквы на другую, пропуском буквы, добавлением лишней буквы, перестановками букв, пропуском дефиса и слитным написанием двух слов.

Замечание. Неоднозначные случаи замены букв, слитного написания слов и пропуска букв не могут быть обработаны на первом этапе правки, например, правописание не или -тся и -ться в глаголах и др.

Примеры:

— Большой прут/большой пруд.

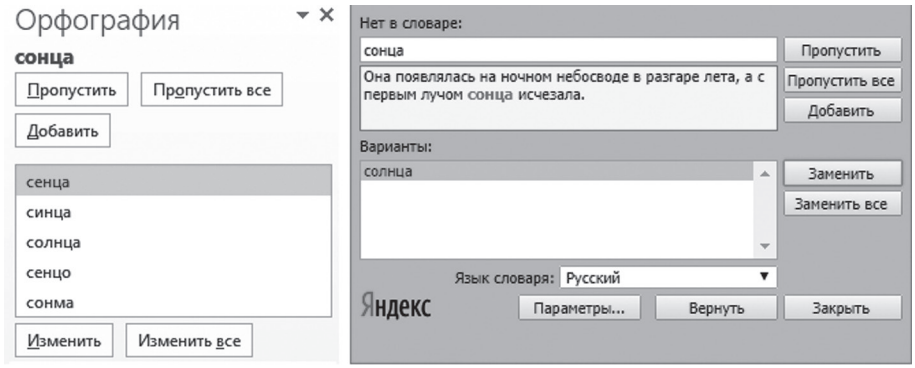
— Сказка ложь/в театре много лож.

— Ох, дяденька, неправда ваша! (В.П. Катаев, Сын полка) — Мы и без него обойдемся — не правда ли? (И.С. Тургенев, Вечер в Сорренте) — Не правда, а ложь всегда режет слух.

Таблица 6

Итоги тестирования

Действие	Word	Яндекс-спеллер	Приложение
Число неисправленных орфографических ошибок			
Выбор первого кандидата/автоматическая правка (этап 1)	22	6	14
Выбор из списка кандидатов (этап 1)	9	6	9
Автоматическая правка (этап 2)	—	—	1
Выбор из списка кандидатов (этап 2)	—	—	0
Число неисправленных грамматических ошибок			
Выбор первогого кандидата/автоматическая правка (этап 1)	19	19	20
Выбор из списка кандидатов (этап 1)	18	18	19
Автоматическая правка (этап 2)	—	—	10
Выбор из списка кандидатов (этап 2)	—	—	2
Всего неисправленных ошибок	27	24	11/2



а *б*

Рис. 4. Проверка правописания с помощью Microsoft Word (*а*) и Яндекс-спеллер (*б*)

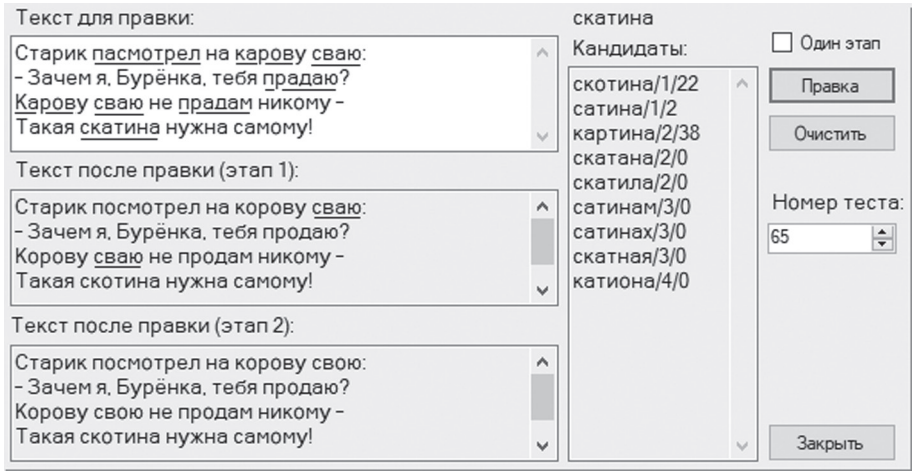


Рис. 5. Форма правки текста:
первая цифра в списке кандидатов — цена замены, вторая — число прецедентов

— Учиться — всегда пригодится (Пословица) — Он хорошо учится.

Подобные случаи, а также другие неисправленные на первом этапе ошибки обрабатываются на втором этапе.

Число оставленных приложением ошибок будет меньше, если пользователю предоставить право выбора замещающего слова из списка кандидатов. Это отражено в строке «Выбор из списка кандидатов» табл. 6. В строке «Всего неисправленных ошибок» цифры 11/2, приведенные для приложения, означают, что при автоматической правке приложение оставит 11 ошибок, если же разрешить интерактивный выбор кандидата, то число неисправленных ошибок сократится до двух.

Пример. При обработке фрагмента «равнодушество убивает» приложением на втором этапе правки был найден прецедент «равнодушие убивает». Более того, слово «равнодушие» оказалось первым в упорядоченном МК, но автоматической правки не произошло, поскольку ЦЗ(равнодушество, равнодушие) = 5, что больше максимально допустимой цены замены (ЦЗМакс = 4).

Проведенное тестирование позволяет сделать вывод о пригодности изложенного подхода для решения задачи правки текстов.

Заключение

Правка текста выполняется в два этапа. Задача первого этапа — освободить текст от орфографических ошибок и ошибок, вызванных неправильным словообразованием. Цель второго — устранить оставшиеся грамматические ошибки.

Рассмотрена база данных, обеспечивающая работу приложения на обоих этапах правки. Для каждого этапа рассмотрены алгоритмы формирования множества кандидатов на замещение слова с ошибкой и выбора лучшего кандидата по критериям «цена замены» и «всего прецедентов». На тестовой выборке приложение исправило 88,8% ошибок в режиме автоматической правки и обеспечило исправление 98% ошибок в режиме интерактивной правки.

В будущем планируется увеличить количество исправляемых на первом этапе ошибок, во-первых, за счет формирования таблицы укороченных слов с расстоянием редактирования 2 и, во-вторых, более полного учета при расчете цены замены фонетической близости слов.

На втором этапе предполагается добавить в базу данных приложения таблицы с векторными представлениями слов, фрагментов и предложений, например, по аналогии с word2vec [13], создать и обучить нейронные сети для решения задач:

— отнесения фрагмента к одному из двух классов: «без грамматических ошибок», «с грамматическими ошибками» (на вход нейронной сети подается векторное представление фрагмента);

— исправления грамматических ошибок во фрагменте, отнесенном ко второму классу (на вход сети подаются векторные представления слов, образующих левый и/или правый контексты исследуемого слова фрагмента, на выходе сети — векторное представление кандидата на замену слова);

— отнесения предложения к одному из двух классов: «без пунктуационных ошибок», «с пунктуационными ошибками» (на вход нейронной сети подается векторное представление предложения);

— исправления пунктуационных ошибок в предложении, отнесенном ко второму классу (на вход нейронной сети подаются векторные представления пары слов, на выходе сети — векторное представление знака препинания между словами или пробела).

Литература

1. **Уроки литературы.** Все виды ошибок [Электрон. ресурс] http://chitaj.ucoz.net/index/vse_vidy_oshibok/0-99 (дата обращения 01.09.2018).

2. **Уроки литературы.** Грамматические ошибки [Электрон. ресурс] http://chitaj.ucoz.net/index/grammaticheskie_oshibki/0-97 (дата обращения 01.09.2018).

3. **Левченко О.С., Тишина Т.Н.** Готовимся к ГИА по русскому языку. 9 класс (пособие для учителя): комментарии, рекомендации, дидактические материалы. Омск: БОУ ДПО «ИРООО», 2009.

4. **Спеллер** [Электрон. ресурс] https://tech.yandex.ru/speller/ (дата обращения 01.09.2018).

5. **Поддержка** системы автоматической правки текста [Электрон. ресурс] http://100byte.ru/stdntswrks/sql/sql.html (дата обращения 01.09.2018).

6. **Титов Д.А.** Методы автоматического поиска и исправления ошибок в предложении [Электрон. ре-

References

1. **Uroki Literatury.** Vse Vidy Oshibok [Elektron. Resurs] http://chitaj.ucoz.net/index/vse_vidy_oshi-bok/0-99 (Data Obrashcheniya 01.09.2018). (in Russian).

2. **Uroki literatury.** Grammaticheskie Oshibki [Elektron. Resurs] http://chitaj.ucoz.net/index/grammaticheskie_oshibki/0-97 (Data Obrashcheniya 01.09.2018). (in Russian).

3. **Levchenko O.S., Tishina T.N.** Gotovimsya k GIA po Russkomu Yazyku. 9 klass (Posobie dlya Uchitelya): Kommentarii, Rekomendatsii, Didakticheskie Materialy. Omsk: BOU DPO «IROOO», 2009. (in Russian).

4. **Speller** [Elektron. Resurs] https://tech.yandex.ru/speller/ (Data Obrashcheniya 01.09.2018). (in Russian).

5. **Podderzhka** Sistemy Avtomaticheskoy Pravki Teksta [Elektron. Resurs] http://100byte.ru/stdntswrks/sql/sql.html (Data Obrashcheniya 01.09.2018). (in Russian).

6. **Titov D.A.** Metody Avtomaticheskogo Poiska i Ispravleniya Oshibok v Predlozhenii [Elektron. Resurs]

сурс] http://100byte.ru/stdntswrks/spellCh/spellCh.html (дата обращения: 01.09.2018).

7. **Garbe W**. 1000x Faster Spelling Correction algorithm (2012). [Электрон. ресурс] https://medium.com/@wolfgarbe/1000x-faster-spelling-correction-algorithm-2012-8701fcd87a5f (дата обращения 01.09.2018).

8. **Варшавский П.Р., Алехин Р.В.** Метод поиска решений в интеллектуальных системах поддержки принятия решений на основе прецедентов // Information Models and Analyses. 2013. V. 2. No. 4. Pp 385—392.

9. **Знаки препинания** [Электрон. ресурс] https://dic.academic.ru/dic.nsf/ruwiki/28257 (дата обращения 01.09.2018).

10. **Фонетический разбор слов.** [Электрон. ресурс] http://phoneticonline.ru/ (дата обращения 01.09.2018).

11. **Словарь** языка Пушкина / Отв. ред. акад. АН СССР В.В. Виноградов. М.: Азбуковник, 2000.

12. **Mikolov T., Sutskever I., Chen K., Corrado G., Dean J.** Distributed Representations of Words and Phrases and their Compositionality // Proc Advances in Neural Information Proc. Syst. 2013. Pp. 3111—3119.

http://100byte.ru/stdntswrks/spellCh/spellCh.html (Data Obrashcheniya: 01.09.2018). (in Russian).

7. **Garbe W.** 1000x Faster Spelling Correction algorithm (2012). [Elektron. resurs] https://medium.com/@wolfgarbe/1000x-faster-spelling-correction-algorithm-2012-8701fcd87a5f (Data Obrashcheniya 01.09.2018).

8. **Varshavskiy P.R., Alekhin R.V.** Metod Poiska Resheniy v Intellektual'nyh Sistemah Podderzhki Prinyatiya Resheniy Na Osnove Pretsedentov. Information Models and Analyses. 2013;2;4:385—392. (in Russian).

9. **Znaki Prepinaniya** [Elektron. Resurs] https://dic.academic.ru/dic.nsf/ruwiki/28257 (Data Obrashcheniya 01.09.2018). (in Russian).

10. **Foneticheskiy Razbor Slov.** [Elektron. Resurs] http://phoneticonline.ru/ (Data Obrashcheniya 01.09.2018). (in Russian).

11. **Slovar'** Yazyka Pushkina. Otv. Red. Akad. AN SSSR V.V. Vinogradov. M.: Azbukovnik, 2000. (in Russian).

12. **Mikolov T., Sutskever I., Chen K., Corrado G., Dean J.** Distributed Representations of Words and Phrases and their Compositionality. Proc Advances in Neural Information Proc. Syst. 2013:3111—3119.

Сведения об авторах:

Бартеньев Олег Васильевич — кандидат технических наук, доцент кафедры прикладной математики НИУ «МЭИ», e-mail: mdf4@mail.ru
Титов Дмитрий Алексеевич — магистрант кафедры прикладной математики НИУ «МЭИ», e-mail: dimnrtyu@mail.ru

Information about authors:

Barten′ev Oleg V. — Ph.D. (Techn.), Assistant Professor ofApplied Mathematics Dept., NRU MPEI, e-mail: mdf4@mail.ru
Titov Dmitriy A. — Master Student of Applied Mathematics Dept., NRU MPEI, e-mail: dimnrtyu@mail.ru

Конфликт интересов: авторы заявляют об отсутствии конфликта интересов
Conflict of interests: the authors declare no conflict of interest

Статья поступила в редакцию: *23.10.2018*
The article received to the editor: *23.10.2018*