Н

УДК 004.89:004.93

А. В. Ниценко, С. А. Большакова, В. Ю. Шелепов
Государственное учреждение «Институт проблем искусственного интеллекта», г. Донецк
83048, г. Донецк, ул. Артема, 118-б

РАЗДЕЛЕНИЕ СПЛОШНОГО ТЕКСТА НА СЛОВА

A. V. Nicenko, S. A. Bolshakova, V. Ju. Shelepov
Public institution «Institute of Problems of Artificial Intelligence», Donetsk
83048, Donetsk, Artema st., 118-b

SPLITTING A SOLID TEXT INTO WORDS

А. В. Ниценко, С. А. Большакова, В. Ю. Шелєпов
Державна установа «Інститут проблем штучного інтелекту», м. Донецьк
83048, м. Донецьк, вул. Артема, 118-б.

ПОДІЛ СУЦІЛЬНОГО ТЕКСТУ НА СЛОВА

В работе предлагается метод разбиения сплошной (без пробелов) строки русскоязычного текста (далее “текст”) на отдельные слова. Метод использует работу со словарем, содержащим большое число русских словоформ и морфологическую информацию о них. Массив словоформ представлен в виде дерева, обеспечивающего быстрый поиск. Выделение первого слова начинается с посимвольного ввода текста от начала до момента, когда результат поиска в дереве совпадет с введенной подстрокой. При продолжении ввода до конца могут возникнуть дополнительные варианты первого слова. Каждый вариант удаляется из текста и аналогично ищется второе слово. Выбор правильного варианта из числа возникающих вариантов разбиения происходит на основе принципа минимума числа слов и удаления вариантов, недопустимых с точки зрения русской грамматики.
Ключевые слова: префиксное дерево, разделение на слова, морфологическая информация

The paper proposes a method for splitting continuous (without spaces) line of Russian-language text into separate words. The method uses work with a dictionary containing a large number of Russian word forms along with their morphological information. The array of word forms represented as a tree that provides a quick search. Selecting the first word starts with a character input from the beginning to the moment when the search result in the tree matches the entered substring. If you continue input to the end, additional variants of the first word may arise. Each variant is removed from the text and the second word is similarly searched. Choosing the correct variant among the arising splitting variants based on the principle of word count minimization and the deletion of variants that does not conform to the rules of Russian grammar.
Key words: prefix tree, word division, morphological information

У роботі пропонується метод розбиття суцільного (без пробілів) рядку російськомовного тексту (далі «текст») на окремі слова. Метод використовує роботу зі словником, що містить велику кількість російських словоформ і морфологічну інформацію про них. Масив словоформ представлений у вигляді дерева, що забезпечує швидкий пошук. Виділення першого слова починається з посимвольного введення тексту від початку до моменту, коли результат пошуку в дереві співпаде з введеним підрядком. При продовженні введення до кінця можуть виникнути додаткові варіанти першого слова. Кожен варіант видаляється з тексту і аналогічно шукається друге слово. Вибір правильного варіанту з числа варіантів розбиття, що виникають, відбувається на основі принципу мінімуму числа слів і видалення варіантів, неприпустимих з точки зору російської граматики.
Ключові слова: префіксне дерево, поділ на слова, морфологічна інформація

Введение

Анализ интернет-источников показывает, что существует интерес к такой проблеме, как разделение сплошного текста (без пробелов) на отдельные слова. Существует необходимость в разумных алгоритмах нахождения вариантов такого разделения и выбора из них правильного. Эта задача может также рассматриваться в качестве модели распознавания слитной речи.

Таким образом, **целью данной работы** является разделение сплошного текста на слова, используя большой словарь русских словоформ. Для решения поставленной цели требуется решить следующие **задачи**:

1) определить метод представления большого словаря словоформ и реализовать алгоритм поиска строки словаря с использованием префиксного дерева;
2) реализовать разбиение слитно написанного текста на слова;
3) определить принципы выбора правильного разбиения, сформулировать правила определения недопустимых последовательностей слов.

Метод использует работу со словарем, содержащим большое число русских словоформ и морфологическую информацию о них. Массив словоформ представлен в виде дерева, обеспечивающего быстрый поиск. Выделение первого слова начинается с посимвольного ввода текста от начала до момента, когда результат поиска в дереве совпадет с введенной подстрокой. При продолжении ввода до конца могут возникнуть дополнительные варианты первого слова. Каждый вариант удаляется из текста и аналогично ищется второе слово. Выбор правильного варианта из числа возникающих вариантов разбиения происходит на основе принципа минимума числа слов и удаления вариантов, недопустимых с точки зрения русской грамматики.

1 Описание работы программы

Мы будем иметь дело с произвольной строкой, которая представляет собой последовательность русских словоформ, набранных сплошным образом, то есть без пробелов. Для выполнения автоматического разбиения ее на отдельные словоформы с помощью пробелов будем использовать обширный словарь русских словоформ, содержащий грамматическую информацию относительно каждой словоформы. Сами словоформы будем называть также словами словаря. Мы будем использовать морфологический словарь М. Хагена [5]. Общий объем словаря составляет около 4 миллионов слов. Каждая строка в применяемом нами варианте состоит из двух полей, разделенных символом «|»: собственно словоформа и морфологическая информация о ней. Морфологическая информация состоит из цепочки сокращений, отражающих часть речи и грамматические характеристики для существительных, глаголов и так далее (например, число, падеж, лицо). Сокращения разделены пробелами, точка в их записи не используется.

В разработанной программе реализовано два режима работы:

1) программа дает множество всех возможных разбиений. Оператор самостоятельно определяет наличие среди них правильного варианта разбиений;

2) программа дает множество возможных разбиений, применяя сформулированные правила определения недопустимых последовательностей слов на основе их морфологической информации. В подавляющем большинстве случаев правильный вариант разбиения будет на первом месте.

Во втором режиме работы разрабатываемая программа обрабатывает не изолированные слова (а именно они рассматривались в качестве входной информации в отдельных работах последнего десятилетия), а учитывает при анализе каждого текстового слова результаты разбора его соседей.

Основное окно реализованной программы, отображающее результаты работы в первом режиме, представлено на рис. 1. А результаты работы во втором режиме представлены на рис. 2 соответственно.

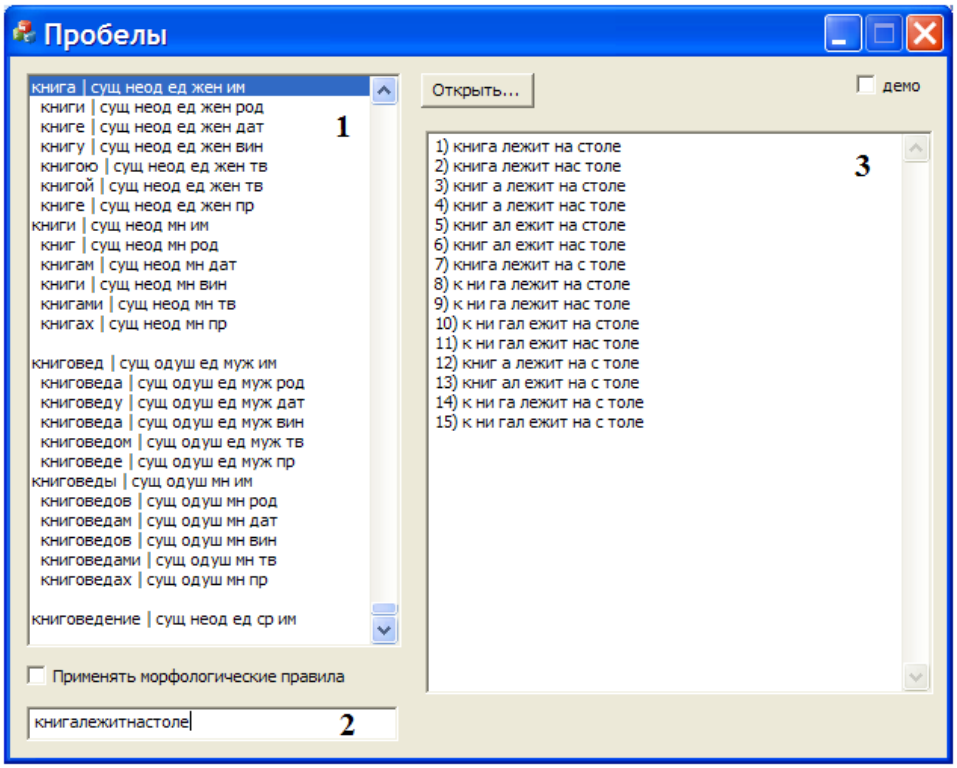

Рисунок 1 – Основное окно программы в первом режиме работы

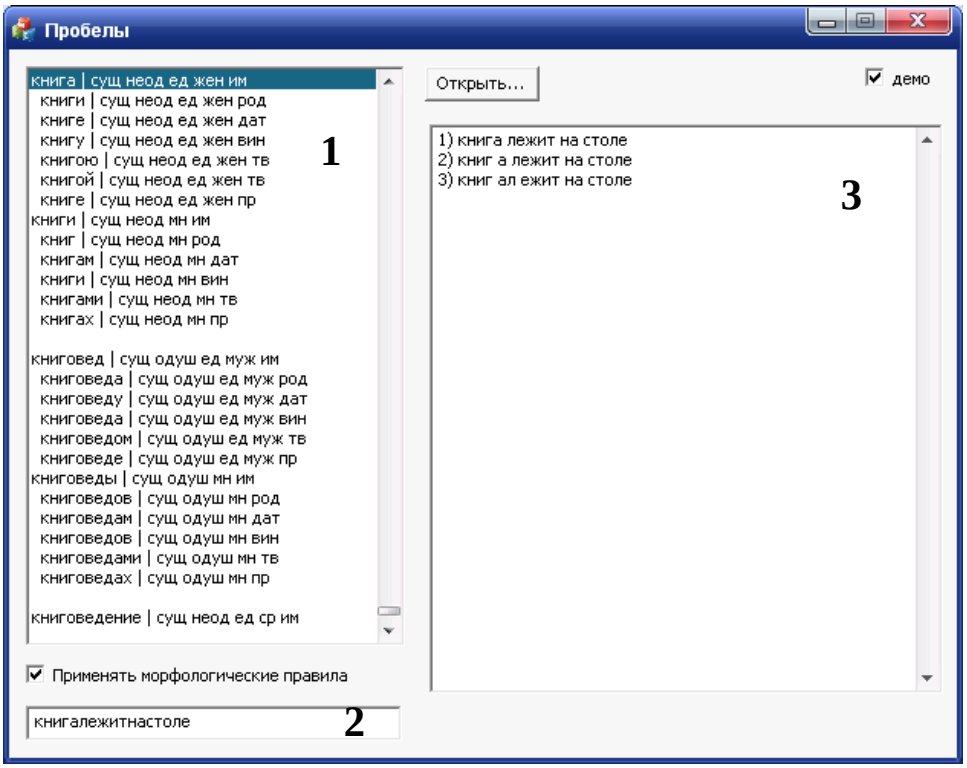

Рисунок 2 – Основное окно программы во втором режиме работы

2 Представление множества словоформ в виде префиксного дерева

Пусть массив словоформ словаря состоит из N словоформ. Для обеспечения быстрого поиска в нем мы будем использовать представление его в виде префиксного дерева, состоящего из множества узлов. В каждом узле дерева хранится метка – один из символов алфавита $A = \{a_1, a_2, ..., a_d\}$.

Ключом, который соответствует некоторому узлу, является путь от корня дерева до узла, а точнее строка $c_1 c_2 ... c_m$, составленная из меток узлов, повстречавшихся на этом пути. Если соответствующая строка есть в словаре, то с узлом ассоциируется индекс, являющийся ее порядковыми номером в словаре. В словаре может быть несколько словоформ с одинаковым написанием, различающихся смысловой или морфологической информацией. Поэтому вместо одного упомянутого индекса может присутствовать некоторый список таких индексов. Наличие индекса или списка индексов указывает на то, что узел является конечным, то есть соответствующим концу некоторого слова. В противном случае узел является лишь промежуточным по дороге в какой-либо другой, который является конечным.

Корень дерева, очевидно, соответствует пустому ключу. Имеет смысл также для каждого узла использовать множество дочерних узлов следующего нижнего уровня (смежные узлы – потомки). Количество их может меняться от *0* до *d*.

Хранение дерева в памяти осуществляется с помощью списка всех его узлов и списка номеров смежных узлов – потомков для каждого из узлов. Для узлов, не имеющих ни одного потомка, список смежных узлов пуст. Если в дереве имеется *L* узлов, пронумерованных 0,…,*L-1*, то в памяти будет храниться *L* списков потомков, собранных в главный список.

0: {1, 5, 9, 15}
1: {2, 4}
2: {3}
…
L-1: {}

Порядковый номер каждого списка смежных узлов соответствует номеру данного узла в списке.

В списке узлов хранятся данные для каждого узла, состоящие из двух полей – символа алфавита и списка индексов строк (для промежуточных узлов последний список будет пустым).

0: [' ',{}]
1: ['а',{0,1,2}]
2: ['б',{}]
3: ['а',{3}]
4: ['ж',{}]
…

3 Алгоритм поиска строки словаря с использованием префиксного дерева

Алгоритм поиска строки словаря с использованием дерева можно описать следующим образом. Пусть дана строка, которую с помощью дерева необходимо найти в словаре иди убедиться в ее отсутствии. Будем спускаться из корня дерева на нижние уровни, каждый раз переходя в узел, чей символ совпадает с очередным

символом строки. После того как обработаны все символы строки, узел, в котором остановился спуск, и будет искомым узлом. Индекс, ассоциированный с этим узлом, – есть номер строки в словаре. Если в процессе спуска не нашлось узла с символом, соответствующим очередному символу строки, или спуск остановился на промежуточной вершине (с пустым списком индексов), то искомый ключ отсутствует в дереве, а строка – в словаре.

Поиск можно наглядно проиллюстрировать в основном окне программы (см. рис. 1) следующим образом. Допустим, ищется словоформа “книга”. Символы, начиная с первого, вводятся один за другим в поле 2. При вводе очередного символа программа считывает получившуюся подстроку и перемещает курсор в поле 1 на строку, содержащую самую короткую словоформу, для которой введенная подстрока является вхождением от начала. Тем самым при вводе словоформы с клавиатуры, мы можем практически мгновенно найти ее в словаре или убедиться в ее отсутствии.

На этом же основан предлагаемый ниже метод поиска вариантов разбиения слитно написанного текста на слова, разделенные пробелами.

В наихудшем случае количество операций сравнения при поиске символа в узле составит $\log_2 d$. Поэтому максимальная сложность поиска строки равна $m \cdot \log_2 d$, где m – длина строки. Таким образом, сложность поиска оказывается пропорциональной длине слова *m*, а не размеру словаря *N*.

4 Разбиение слитно написанного текста на слова

Вариантов выделения первого, второго и так далее слов может быть несколько. Если в каждом случае такой вариант один, выделение этих слов можно проиллюстрировать в основном окне программы (рис.1) следующим образом. Начнем вводить наш текст в поле 2, посимвольно наращивая подстроку ввода. Тогда, как описано выше, курсор в поле 1 будет перемещаться по некоторым словам словаря. Отслеживаем, когда выделенное в поле 1 слово совпадет с введенной подстрокой. Это и будет искомое первое слово. Далее удаляем выделенное первое слово из нашего слитного текста и, применяя описанную процедуру к остатку, выделяем второе слово и так далее. Запись последовательно выделенных слов через пробел дает в этом случае решение поставленной задачи. Для автоматической обработки слитного текста ввод с клавиатуры и упомянутое выше отслеживание должны быть имитированы программно.

Для более формального описания алгоритма, охватывающего случай, когда на некоторых этапах выделения последовательных слов возникает более одного варианта, используем обозначения:

Text – Текст, не разделенный пробелами (строка без пробелов),

L – Список слов вместе с соответствующей им морфологической информацией (в начале работы алгоритма – пуст),

S – Список разбиений (в начале работы алгоритма – пуст).

Алгоритм использует рекурсивное применение следующей функции:

AutoSpace (L, Text):

{

1) Определение W – списка возможных слов W[0],…,W[N-1], с которых может начинаться не разделенная пробелами строка Text, и которые присутствуют в словаре, с помощью последовательного наращивания строки, начиная с первого символа и поиска получившихся подстрок в словаре. Поиск завершается, если достигнут последний символ в тексте, либо если для текущей

подстроки в дереве невозможно найти соответствующий путь. Если не найдено ни одного слова (список W пуст), то выход из функции AutoSpace.

2) Цикл по *i* от *0* до *N-1*

Начало цикла

Копируем Text в TextCopy;

Удаляем W[i] из TextCopy;

Копируем L в новый список L_Copy;

Добавляем W[i] вместе с соответствующей морфологической информацией в L_Copy;

Если TextCopy не пусто, то:

Вызов AutoSpace (L_Copy, TextCopy);

Иначе:

Добавляем L_Copy в список возможных разбиений S.

Конец цикла

3) выход из функции AutoSpace;

}

Далее осуществляется вывод списка *S* слов, разделенных пробелами.

В качестве примера приведем работу с текстом «книгалежит». Начальная реализация пункта 1) функции AutoSpace дает 3 варианта выделения 1-го слова: «к», «книг» и «книга». Рисунки 3 – 5 иллюстрируют дальнейшие результаты работы функции AutoSpace в каждом из этих случаев.

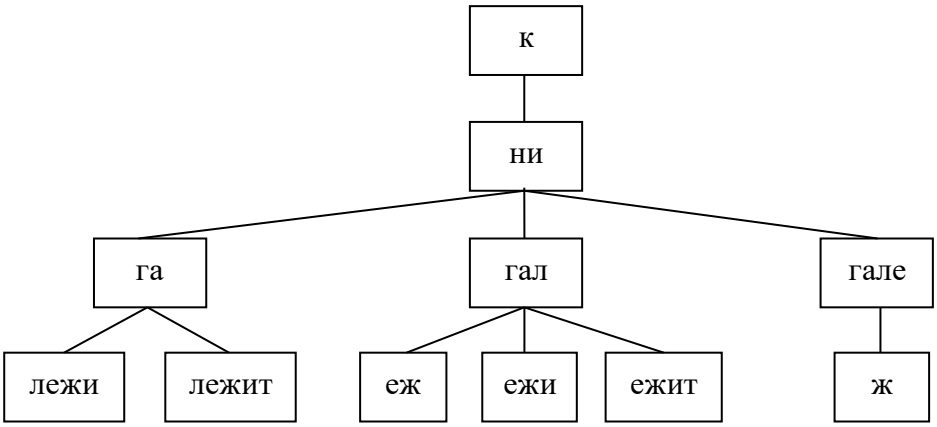

Рисунок 3 – Результаты работы функции AutoSpace при первом слове «к»

Здесь «гал» единица измерения ускорения, названная так по имени Галилея; «еж», «ежит» – слова, которые при более педантичной записи были бы написаны через «ё».

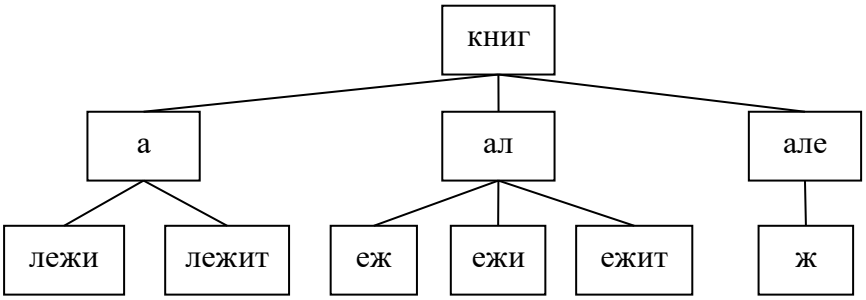

Рисунок 4 – Результаты работы функции AutoSpace при первом слове «книг»

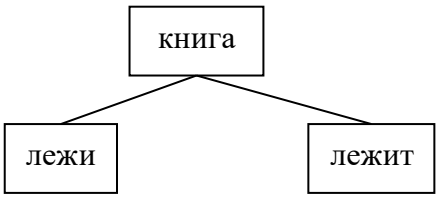

Рисунок 5 – Результаты работы функции AutoSpace при первом слове «книга»

Таким образом, получаются следующие варианты разбиения сплошного текста на слова:

к ни га лежит
к ни гал ежит
книг а лежит (1)
книг ал ежит
книга лежит

5 Работа с вариантами разбиения

В общем случае число вариантов даже для короткого предложения, написанного без пробелов, может исчисляться сотнями. Возникает вопрос: как из множества вариантов разбиения выбрать правильный? Для этого, прежде всего, предлагается упорядочить варианты по возрастанию количества слов или, что то же самое, по количеству пробелов. В нашем примере применение сортировки с этим критерием дает следующий список:

книга лежит
к ни гал ежит
книг а лежит
книг ал ежит
к ни га лежит

Здесь правильный вариант, выделенный жирным шрифтом, стоит на первом месте. Многочисленные эксперименты показывают, что можно сформулировать некий принцип минимума: *ПРИ УПОРЯДОЧЕНИИ ВАРИАНТОВ ПО КОЛИЧЕСТВУ СЛОВ ПРАВИЛЬНЫЙ ВАРИАНТ НАХОДИТСЯ БЛИЗКО К НАЧАЛУ СПИСКА.* В подавляющем большинстве случаев он просто является первым.

В связи с описанной ситуацией актуальными являются шаги по сокращению числа вариантов на основании морфологической информации относительно слов разбиения.

Для приведенных выше разбиений (1) информация о морфологической характеристике слов представлена в табл. 1.

Далее приведены некоторые примеры правил, по которым определяются недопустимые варианты. Пусть рассматриваемый вариант состоит из *К* слов, пронумерованных от *0* до *К -1*.

1) предложение не может заканчиваться предлогом или союзом:
Слово[К-1].ЧастьРечи != «предл»
Слово[К-1].ЧастьРечи != «союз»
2) предлог не может стоять перед глаголом или союзом:
Слово[i].ЧастьРечи == «предл» И Слово[i+1].ЧастьРечи != «гл»
Слово[i].ЧастьРечи == «предл» И Слово[i+1].ЧастьРечи != «союз»

3) не допускается употребление двух предлогов подряд (за исключением составных предлогов):

Слово[i].ЧастьРечи == «предл» И Слово[i+1].ЧастьРечи != «предл»

Таблица 1 – Слова из разбиений и их морфологические характеристики

к | предл дат	**ни** | союз **ни** | част	**га** | сущ неод ед муж **га** | межд	**лежит** | гл несов непер наст ед 3-е
к | предл дат	**ни** | союз **ни** | част	**гал** | сущ неод ед муж им	**ежит** | гл несов непер наст ед 3-е
книг | сущ неод мн род	**а** | союз **а** | част **а** | межд		**лежит** | гл несов непер наст ед 3-е
книг | сущ неод мн род	**ал** | прл крат ед муж		**ежит** | гл несов непер наст ед 3-е
книга | сущ неод ед жен ин	**лежит** | гл несов непер наст ед 3-е		

Если несколько найденных вариантов разбиения состоят из словоформ, совпадающих по написанию, но различающихся грамматическими свойствами, то оставляется только один из них.

Работа описанной системы предполагает некоторое участие пользователя. Список вариантов, выводимый в поле 3 (рис.1) пронумерован. Если правильный вариант является *n*-ым по счету, то после нажатия пользователем цифры *n* на клавиатуре этот вариант остается в поле 3 единственным. Если правильный вариант – первый, то вместо 1 можно нажать Enter. Возможна иная организация работы, когда автоматически выводится только первый вариант, а остальные выводятся по нажатии кнопки Esc, после чего пользователь, нажав цифру, выберет нужный вариант.

Заключение

Разработаны метод и алгоритмы для разделения сплошного текста на слова. Они реализованы как компьютерная программа, которая работает с минимальным участием пользователя. Для разработки этой программы были сформулированы правила определения недопустимых последовательностей слов на основе их морфологической информации. Они могут являться самоценной информацией, которая может быть использована научным сообществом и для исследовательских целей, и для улучшения эффективности прикладных разработок.

Предложенный метод применим не только к русскому, но и к другим языкам. Разумеется, в каждом случае понадобится модификация, связанная, например, с другими механизмами синтаксического управления.

Список литературы

1. Саввина Г. В. Распознавание ключевых слов в потоке слитной речи [Текст] / Г. В. Саввина // Искусственный интеллект. – 2000. – № 3. – С. 543–551.
2. Шишков Г. П. Сегментация слитного текста от слитной речи [Электронный ресурс] / Г. П. Шишков // Тр. Международного семинара Диалог-2001 по компьютерной лингвистике и ее приложениям. 2001. URL: http://www.dialog-21.ru/digest/2001/articles/shishkov/ (дата обращения: 27.11.2018).

3. Sudarshan S. C. Lexical text segmentation using dictionaries [Электронный ресурс] / S. C. Sudarshan // IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). 2018. – URL : https://ieeexplore.ieee.org/document/8301687/ (дата обращения: 27.11.2018).
4. Van Aken J. A statistical learning algorithm for word segmentation [Электронный ресурс] / Van Aken J. // Microsoft Corporation. 2011. – URL: https://arxiv.org/ftp/arxiv/papers/1105/1105.6162.pdf (дата обращения: 27.11.2018).
5. Хаген М. Полная парадигма. Морфология [Электронный ресурс] / М. Хаген // Форум «Говорим по-русски» [сайт]. 2018. – URL: http://www.speakrus.ru/dict/#morph-paradigm (дата обращения: 19.11.2018)
6. Шелепов В. Ю. О распознавании первого звука в слитном речевом отрезке [Текст] / В. Ю. Шелепов, А. В. Ниценко // Проблемы искусственного интеллекта – 2015. – № 0(1). – С. 116–122.
7. Шелепов В. Ю. О распознавании русских слов с использованием обобщенной транскрипции [Текст] / В. Ю. Шелепов, А. В. Ниценко // Проблемы искусственного интеллекта – 2018. – № 1(8). – С. 50–56.

References

1. Savvina G.V. Raspoznavanie klyuchevyih slov v potoke slitnoy rechi [Keyword recognition in a stream of continuous speech]. *Iskusstvennyiy intellekt* [Artificial Intelligence], 2000, no. 3, pp. 543-551.
2. Shishkov G.P. Segmentatsiya slitnogo teksta ot slitnoy rechi [Segmentation of a continuous text from a continuous speech]. *Tr. Mejdunarodnogo seminara Dialog-2001 po kompyuternoy lingvistike i ee prilojeniyam* [Proc. International Seminar Dialogue 2001 on computational linguistics and its applications]. 2001. Available at: http://www.dialog-21.ru/digest/2001/articles/shishkov/ (Accessed 27.11.2018).
3. Sudarshan S.C. Lexical text segmentation using dictionaries. *IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. 2018. Available at: https://ieeexplore.ieee.org/document/8301687/ (Accessed 27.11.2018).
4. Van Aken J. A statistical learning algorithm for word segmentation. *Microsoft Corporation. 2011*. Available at: https://arxiv.org/ftp/arxiv/papers/1105/1105.6162.pdf (Accessed 27.11.2018).
5. Hagen M. Polnaya paradigma. Morfologiya [The complete paradigm. Morphology]. *Forum «Govorim po-russki»* [Forum "Speak Russian"]. 2018. Available at: http://www.speakrus.ru/dict/#morph-paradigm (Accessed 19.11.2018)
6. Shelepov V.Ju., Nicenko A.V. O raspoznavanii pervogo zvuka v slitnom rechevom otrezke [On Recognition of the First Sound in Continuous Speech Fragment]. *Problems of Artificial Intelligence*, 2015, no. 0(1), pp. 116 – 122.
7. Shelepov V.Ju., Nicenko A.V. O raspoznavanii russkikh slov s ispol'zovaniyem obobshchennoy transkriptsii [On the recognition of russian words using generalized transcription]]. *Problems of Artificial Intelligence*, 2018, no. 1(8), pp. 50 – 56.

RESUME

A. V. Nicenko, S. A. Bolshakova, V. Ju. Shelepov

Splitting a solid text into words

Background: An analysis of Internet materials shows that there is interest in the problem of splitting a solid text (without spaces) into individual words. Rational algorithms required to find options for such splitting and to choose the correct one. This task can also be considered as a model for recognition of continuous speech.

Materials and methods: The method uses work with a dictionary containing a large number of Russian word forms along with their morphological information. The array of word forms represented as a tree that provides a quick search. Selecting the first word starts with a character input from the beginning to the moment when the search result in the tree matches the entered substring. If you continue input to the end, additional variants of the first word may arise. Each variant is removed from the text and the second word is similarly searched. Choosing the correct variant among the arising splitting variants based on the principle of word count minimization and the deletion of variants that does not conform to the rules of Russian grammar.

Results: A method and algorithms for splitting a solid text into words have been developed. They are implemented as a computer program that works with minimal user participation.

Conclusion: The proposed method is applicable not only to Russian, but also to other languages. Of course, in each case a modification is required associated, for example, with other mechanisms of syntactic control.

РЕЗЮМЕ

А. В. Ниценко, С. А. Большакова, В. Ю. Шелепов

Разделение сплошного текста на слова

Предпосылки: Анализ интернет-источников показывает, что существует интерес к такой проблеме, как разделение сплошного текста (без пробелов) на отдельные слова. Существует необходимость в разумных алгоритмах нахождения вариантов такого разделения и выбора из них правильного. Эта задача может также рассматриваться в качестве модели распознавания слитной речи.

Материалы и методы: Метод использует работу со словарем, содержащим большое число русских словоформ и морфологическую информацию о них. Массив словоформ представлен в виде дерева, обеспечивающего быстрый поиск. Выделение первого слова начинается с посимвольного ввода текста от начала до момента, когда результат поиска в дереве совпадет с введенной подстрокой. При продолжении ввода до конца могут возникнуть дополнительные варианты первого слова. Каждый вариант удаляется из текста и аналогично ищется второе слово. Выбор правильного варианта из числа возникающих вариантов разбиения происходит на основе принципа минимума числа слов и удаления вариантов, недопустимых с точки зрения русской грамматики.

Результаты: Разработаны метод и алгоритмы для разделения сплошного текста на слова. Они реализованы как компьютерная программа, которая работает с минимальным участием пользователя.

Заключение: Предложенный метод применим не только к русскому, но и к другим языкам. Разумеется, в каждом случае понадобится модификация, связанная, например, с другими механизмами синтаксического управления.

Статья поступила в редакцию 10.09.2018.