УДК 004.93
ББК 32.972.1

ОБЗОР АЛГОРИТМОВ ФОНЕТИЧЕСКОГО КОДИРОВАНИЯ

Выхованец В. С.[1]**,**
(*ФГБУН Институт проблем управления им. В.А. Трапезникова РАН, Москва)*
Ду Ц.[2]**, Сакулин С. А.**[3]
(*МГТУ им. Н.Э. Баумана, Москва*)

Приведен обзор алгоритмов фонетического кодирования, предназначенных для определения схожести слов по звучанию (произношению). Алгоритмы фонетического кодирования разделены на алгоритмы для сравнения слов и алгоритмы определения расстояния между словами. Описаны алгоритмы сравнения слов SoundEx, NYSIIS, Daitch-Mokotoff, Metaphone, Polyphone и алгоритмы вычисления расстояния между словами Левенштейна, Джаро, на основе N-грамм. Для каждого алгоритма указаны его достоинства и недостатки, приводится аналог алгоритма для русского языка. Для устранения общих недостатков алгоритмов фонетического кодирования предложено использовать не последовательности букв слов, а последовательности их элементарных звуков.

Ключевые слова: алгоритм фонетического кодирования, фонетическое расстояние, индексирование слов по звучанию.

---

[1] *Валерий Святославович Выхованец, доктор технических наук, доцент* (*valery@vykhovanets.ru*).
[2] *Цзяньмин Ду, аспирант* (*forsola@qq.com*).
[3] *Сергей Александрович Сакулин, кандидат технических наук, доцент* (*ss141291@yandex.ru*).

1. Введение

Алгоритмы фонетического кодирования представляют собой алгоритмы индексирования слов по их звучанию, которые на основе последовательности букв слова и правил произношения преобразуют их в кодирующий текст (код, индекс, ключ). При совпадении или близости кодирующего текста для двух различных слов делается вывод о близости этих слов по звучанию.

Первым представителем алгоритмов фонетического кодирования для английского языка стал алгоритм SoundEx [69], который использовался в тридцатые годы для кодирования фамилий при переписи населения. В алгоритме используется метод кодирования, призванный устранить орфографические и типографические ошибки в именах. Например, алгоритм SoundEx вычислит одинаковый кодирующий текст «S530» для таких слов как «Smith», «Smithe» и «Smyth», т.е. отождествит эти слова как одинаковые по звучанию.

С появлением и развитием компьютерных технологий появилось множество других алгоритмов фонетического кодирования, в том числе и для различных естественных языков. В качестве примеров такого рода алгоритмов следует назвать алгоритмы NYSIIS [77], улучшенный SoundEx [75], Metaphone [40] и др.

Алгоритмы фонетического кодирования включают в себя не только алгоритмы для сравнения слов, но и алгоритмы определения расстояния между словами при поиске по звучанию. На практике наибольшее распространение получили алгоритмы вычисления расстояния Левенштейна [2] и Джаро [30], а также расстояния на основе *N*-грамм [33].

Алгоритмы фонетического кодирования широко используются в областях где требуется сопоставление акустических данных с текстовыми образцами. Например, в области распознавания речи, для проверки написания и исправления слов, для поиска в базах данных, при добыче полезных ископаемых, для идентификации пользователей, для кросс-языкового кодирования речи, при поиске в сети интернет и т.д.

Несмотря на появление и развитие новых моделей и методов распознавания речи, таких как вероятностные и статистические алгоритмы, скрытые марковские модели, нейронные сети, машинное обучение и др., алгоритмы фонетического кодирования не утратили своей актуальности, так как являются базовыми для применения этих моделей и методов на практике [14, 22, 44].

Настоящая статья посвящена обзору алгоритмов фонетического кодирования, главная цель которого – показать те решения, которые используются для сравнения строк по звучанию, а также выявить тенденции их развития.

2. Базовые алгоритмы

2.1. АЛГОРИТМ SOUNDEX

Алгоритм SoundEx запатентован почти сто лет назад [69]. Алгоритм SoundEx для английского языка состоит из пяти шагов.

Шаг 1. Сохранить первую букву слова.

Шаг 2. Удалить из слова буквы A, E, I, O, U, Y, W и H.

Шаг 3. Заменить оставшиеся буквы на цифры согласно таблице 1.

Таблица 1. Кодирование букв в алгоритме SoundEx

Цифра	Буквы
1	B, P, F, V
2	C, S, K, G, J, Q, X, Z
3	D, T
4	L
5	M, N
6	R

Шаг 4. Если в коде имеются группа из одинаковых цифр, то заменить эту группу первой цифрой, за исключением цифр, разделяющих буквы W и H в исходном слове.

Шаг 5. Сформировать результирующий код из первой буквы слова и трех первых цифр, полученный на предыдущих шагах. Если в коде менее трех цифр, то дополнить код нулями.

Пример 1. Рассмотрим следующие два слова: «Lee» и «Shaw». Применив алгоритм, описанный выше, получаем кодирующий текст «L000» для слова «Lee» и «S000» – для слова «Shaw».

Пример 2. Для пояснения шага 4 рассмотрим слово «Ashcraft». Из-за наличия шага 4 кодирующий текст для этого слова будет «A226», а не «A261».

Как видно из таблицы 1, основной принцип кодирования букв в алгоритме состоит в том, что близкие по звучанию буквы кодируются одной и той же цифрой, а непроизносимые буквы удаляются. Однако алгоритм не свободен от недостатков.

Первый недостаток состоит в том, что существуют близкие по звучанию слова, которые имеют разный кодирующий текст.

Пример 3. Для слов «Lee» и «Leigh», имеющих одно и то же звучание, получаем различные коды: «L000» и «L200» соответственно.

Второй недостаток обратный первому: существуют различные по звучанию слова, которые имеют одинаковый кодирующий текст.

Пример 4. Слова «Gauss» и «Ghosh» с разным звучанием имеют код «G200».

Для частичного устранения проблем, выявленных в процессе эксплуатации алгоритма, в настоящее время для английского языка используется модифицированный алгоритм SoundEx со следующей таблицей кодирования.

Таблица 2. Кодирование букв в модифицированном алгоритме

Цифра	Буквы
1	B, P
2	F, V
3	C, S, K
4	G, J
5	Q, X, Z

6	D, T
7	L
8	M, N
9	R

Из таблицы 2 видно, что из групп 1 и 2 исходного алгоритма образованы новые группы со своим цифровым кодом. Дополнительно к этому длина кодирующего текста в модифицированном алгоритме не ограничивается четырьмя знаками.

Как показывают эксперименты, на одно значение кода SoundEx приходится до 21 фамилии. У модифицированного алгоритма таких фамилий две или три.

Алгоритм SoundEx сильно зависит от языка. Поэтому разработано множество модификаций этого алгоритма для различных языков: китайского [20], испанского [7], персидского [36], арабского [56], малайского [53] и т.д. Для русского языка используется кодирование букв в соответствии с таблицей 3, а удаляемыми буквами являются буквы У, Е, Ё, Ы, А, О, Э, Я, И, Ю, Ь, Ъ.

Таблица 3. Кодирование русских букв в алгоритме SoundEx

Цифра	Буквы
1	Б, П
2	Ф, В
3	Ж, З, С, Х
4	К, Г,
5	Ц, Ч, Ш, Щ
6	Д, Т
7	Л, Й
8	М, Н
9	Р

Несмотря на свои недостатки, алгоритм SoundEx имеет положительные отзывы [37]. В настоящее время алгоритм широко используется для сравнения слов по звучанию и значительно повышает вероятность идентификации слов. Благодаря своей

простоте и низкой вычислительной сложности алгоритм SoundEx стал стандартом и встроен в механизм поиска почти всех известных систем управления базами данных [27].

2.2. АЛГОРИТМ NYSIIS

Алгоритм NYSIIS разработан в 1970 году для использования в одноименной информационной системе «New York State Identification and Intelligence System» [77]. Этот алгоритм дает несколько лучшие результаты по сравнению с алгоритмом SoundEx, используя более сложные правила преобразования исходного слова в результирующий код.

Алгоритм учитывает произношение слов английского языка и состоит из шести шагов.

Шаг 1. Преобразование префикса (начала слова) путем следующих подстановок: MAC → MCC; KN → N; K → C; PH, PF → FF; SCH → SSS.

Шаг 2. Преобразование суффикса (конца слова) путем следующих подстановок: EE → Y; IE → Y; DT, RT, RD, NT, ND → D.

Шаг 3. Преобразование слова в целом путем следующих подстановок: EV → AF; A, E, I, O, U → A; Q → G; Z → S; M → N; KN → N; K → C; SCH → SSS; PH → FF, W → A.

Шаг 4. Удаление H после гласных и S, A в конце слова.

Шаг 5. Преобразование суффикса слова путем подстановки AY → Y.

Шаг 6. Ограничить полученный код 6 знаками.

Как видно из описания алгоритма, в NYSIIS используется большое число правил, связывающих написание с произношением, а также учитывается функционирование гласных звуков при произношении слов (все гласные заменяются гласной A).

При сравнительном исследовании алгоритмов фонетическом кодирования установлено, что алгоритм NYSIIS более всего подходит для кодирования фамилий, где дает отличные результаты.

Пример 5. Фамилии «Brain», «Brown» и «Brun» имеют код «Bran», фамилии «Capp», «Cope», «Copp» и «Kipp» – код «Cap»,

фамилии «Dane», «Dean», «Dent» и «Dionne» – код «Dan», фамилии «Smith», «Schmit», «Schmidt» – код «Snat», фамилии «Trueman», «Truman» – код «Tranan» [27].

2.3. АЛГОРИТМ DAITCH-MOKOTOFF SOUNDEX

В процессе использования фонетических алгоритмов установлено, что алгоритмы SoundEx и NYSIIS плохо работают со словами других языков. Для учета специфики произношения слов в других языках разработан алгоритм Daitch-Mokotoff SoundEx, названный по фамилиям авторов, который имеет такие усовершенствования как использование большей длины кода и учет различных произношений слов, выражающийся во множественном кодировании одного и того же слова.

Алгоритм имеет более сложные правила преобразования исходного слова в его код. Как и у алгоритма NYSIIS, в формировании результирующего кода участвуют не только одиночные буквы, но и их последовательности. Преобразования слова в числовой код осуществляются по следующей таблице [75]

Таблица 4. Кодирование букв в улучшенном алгоритме SoundEx

Последовательность букв	Н	Г	О
AI, AJ, AY, EI, EY, EJ, OI, OJ, OY, UI, UJ, UY	0	1	
AU	0	7	
IA, IE, IO, IU	1		
EU	1	1	
A, UE, E, I, O, U, Y	0		
J	1	1	1
SCHTSCH, SCHTSH, SCHTCH, SHTCH, SHCH, SHTSH, STCH, STSCH, STRZ, STRS, STSH, SZCZ, SZCS	2	4	4
SHT, SCHT, SCHD, ST, SZT, SHD, SZD, SD	2	43	43
CSZ, CZS, CS, CZ, DRZ, DRS, DSH, DS, DZH, DZS, DZ, TRZ, TRS, TRCH, TSH, TTSZ, TTZ, TZS, TSZ, SZ, TTCH, TCH, TTSCH, ZSCH, ZHSH, SCH, SH, TTS, TC, TS, TZ, ZH, ZS	4	4	4
SC	2	4	4

DT, D, TH, T	3	3	3
CHS, KS, X	5	54	54
S, Z	4	4	4
CH, CK, C, G, KH, K, Q	5	5	5
MN, NM	66	66	66
M, N	6	6	6
FB, B, PH, PF, F, P, V, W	7	7	7
H	5	5	
L	8	8	8
R	9	9	9

Порядок преобразований соответствует порядку последовательностей букв в таблице 4. Колонки Н, Г и О задают цифровые коды для последовательности букв из первой колонки: Н – в начала слова, Г – перед гласной, О – в остальных случаях.

Альтернативные коды слова, учитывающие различные произношения, поучаются для слов, образованных из исходного путем следующих подстановок: CH → KH, TCH; CK → K, TSK; C → K, TZ; J → Y, DZH; RS → RTZ, ZH.

Пример 6. Имя «Peters» с SoundEx кодом «P362» при использовании алгоритма Daitch-Mokotoff SoundEx получает два кода: 739400 для формы «Peters» и 734000 для формы «Petertz», а имя «Jackson» с SoundEx кодом «J250» – четыре кода: 154600, 454600, 145460, 445460 для форм «Jackson», «Yakson», «Jakson» и «Jatskon» соответственно.

2.4. АЛГОРИТМ METAPHONE

Metaphone – еще один алгоритм фонетического кодирования слов с учётом основных правил английского языка, разработанный в 1990 г. [40]. Он отличается от предыдущих алгоритмов тем, что реализует более сложные правила преобразования. Другое отличие состоит в том, что буквы не разбиваются на группы и не кодируются цифрами. На выходе алгоритм даёт код переменной длины, состоящий из букв.

Алгоритм включает в себя 16 шагов [10].

Шаг 1. Удаление повторяющихся соседних букв кроме буквы С.

Шаг 2. Преобразования префикса слова путем следующих подстановок: KN → N; GN → N; PN → N; AE → E; WR → R.

Шаг 3. Удаление суффикса MB.

Шаг 4. Преобразования последовательности букв с буквой C путем следующих подстановок: CIA → XIA; SCH → SKH; CH → XH; CI → SI; CE → SE; CY → SY; C → K.

Шаг 5. Преобразования последовательности букв с буквой D путем следующих подстановок: DGE → JGE; DGY → JGY; DGI → JGY; D → T.

Шаг 6. Подстановка GH → H, если GH стоит не в конце слова и не перед гласной.

Шаг 7. Подстановка суффиксов GN → N и GNED → NED.

Шаг 8. Преобразования последовательности букв с буквой G путем следующих подстановок: GI → JI; GE → JE; GY → JY; G → K.

Шаг 9. Удаление H после гласных, но не перед гласными.

Шаг 10. Преобразования слова в целом путем следующих подстановок: CK → K; PH → F; Q → K; V → F; Z → S.

Шаг 11. Преобразования последовательности букв с буквой S путем следующих подстановок: SH → XH; SIO → XIO; SIA → XIA.

Шаг 12. Преобразования последовательности букв с буквой T путем следующих подстановок: TIA → XIA; TIO → XIO; TH → 0; TCH → CH.

Шаг 13. Подстановка префикса WH → W. Если после W нет гласной, то удаление W.

Шаг 14. Подстановка префикса X → S, в середине слова – X → KS.

Шаг 15. Удаление Y, которые не находятся перед гласными.

Шаг 16. Удаление всех гласных, кроме начальной.

Пример 7. Фамилии «Brain», «Brown» и «Brun» имеют код «BRN», фамилии «Capp», «Cope», «Copp» и «Kipp» – код «KP», фамилии «Dane», «Dean» и «Dionne» – код «TN», но фамилия «Dent» – код «TNT» (см. пример 5), фамилия «Smith» – код «SM0», фамилия «Schmit» – код «SXMT», фамилия «Schmidt» – код «SXMTT», фамилии «Trueman», «Truman» – код «TRMN».

Позднее в 2000 г. была разработана вторая версия алгоритма, которая получила название Double Metaphone [41], где, в отличие от первой версии, применимой только к английскому языку, учитываются особенности произношения слов, заимствованных из других языков [41]. Для этих слов результатом работы алгоритма являются два кода – по одному для каждого варианта произношения.

Хотя Double Metaphone имеет преимущества перед алгоритмом Metaphone, он по-прежнему имеет некоторые ограничения [45]. В частности, все так же встречаются слова с разным произношением и одинаковым кодом, например, слова «Alice» «Elsa» и «Ullos» кодируются как «ALS».

В 2009 г. появилась третья коммерческая версия алгоритма под названием Metaphone 3. Утверждается, что новый алгоритм увеличивает точность отождествления слов с 89% – Double Metaphone, до 98 % – Metaphone 3 [58]. Также Metaphone 3 начал поддерживать заимствованные слова из большего числа языков. Алгоритм Metaphone 3 достаточно сложный и включает большое число правил. Описание этого алгоритма на языке Java занимает более семи тысяч строк [42].

Алгоритм Metaphone был адаптирован к русскому языку [1]. Для русского языка алгоритм Metaphone состоит из пяти шагов.

Шаг 1. Преобразование гласных путем следующих подстановок: О, Ы, Я → А; Ю → У; Е, Ё, Э, ЙО, ЙЕ → И.

Шаг 2. Оглушение согласный букв, за которыми следует любая согласная, кроме Л, М, Н или Р, либо согласных на конце слова путем следующих подстановок: Б → П; З → С; Д → Т; В → Ф; Г → К.

Шаг 3. Удаление повторяющихся букв.

Шаг 4. Преобразование суффикса слова путем следующих подстановок: УК, ЮК → 0; ИНА → 1; ИК, ЕК → 2; НКО → 3; ОВ, ЕВ, ИЕВ, ЕЕВ → 4; ЫХ, ИХ → 5; АЯ → 6; ЫЙ, ИЙ → 7; ИН → 8; ОВА, ЕВА, ИЕВА, ЕЕВА → 9; ОВСКИЙ → @; ЕВСКИЙ → #; ОВСКАЯ → $; ЕВСКАЯ → %.

Шаг 5. Удаление букв Ъ, Ь и дефиса.

Однако из-за небольшого числа правил Metaphone для русского языка не отождествляет некоторые фонетические схожие слова.

2.5. АЛГОРИТМ POLYPHONE

Специально для русского языка разработан алгоритм фонетического кодирования Polyphone [59]. При разработке алгоритма учтены морфологические, фонологические, фонетические и исторические аспекты произношения слов в русском языке.

Алгоритм состоит из следующих шагов.

Шаг 1. Подстановка вместо латинских букв схожих букв русского алфавита: A → А; E → Е; O → О; C → С; X → Х; B → В; M → М; H → Н. Последние три подстановки применяются только для прописных букв.

Шаг 2. Удаление всех букв, не принадлежащих алфавиту русского языка.

Шаг 3. Удаление букв Ь, Ъ.

Шаг 4. Замена двух одинаковых букв одной.

Шаг 5. Замена одиночных букв: А, Е, Ё, И, О, Ы, Э, Я → А; Б → П; В → Ф; Г → К; Д → Т; З → С; Щ → Ш; Ж → Ш; М → Н; Ю → У.

Шаг 6. Преобразование слова путем выполнения следующих подстановок: АКА → АФА; АН → Н; ЗЧ → Ш; ЛНЦ → НЦ; ЛФСТФ → ЛСТФ; НАТ → Н; НТЦ → НЦ; НТ → Н; НТА → НА; НТК → НК; НТС → НС; НТСК → НСК; НТШ → НШ; ОКО → ОФО; ПАЛ → ПЛ; РТЧ → РЧ; РТЦ → РЦ; СП → СФ; ТСЯ → Ц; СТЛ → СЛ; СТН → СН; СЧ → Ш; СШ → Ш; ТАТ → Т; ТСА → Ц; ТАФ → ТФ; ТС → ТЦ; ТЦ → Ц; ТЧ → Ч; ФАК → ФК; ФСТФ → СТФ; ШЧ → Ш.

Пример 8. В соответствии с описанным выше алгоритмов слова «ТЕЛЕГРАММА» и «АППАРАТ» преобразуются в «ТАЛАКРАМА» и «АПАРАТ».

Для нечеткого сравнения слов предлагается использовать сумму простых чисел, закрепленных за каждой буквой преобразованного слова: А → 2; П → 3; К → 5; Л → 7; М → 11; Н → 13;

Р → 17; С → 19; Т → 23; У → 29; Ф → 31; Х → 37; Ц → 41; Ч → 43; Щ → 47; Э → 53; Я → 59.

Пример 9. Слова из предыдущего примера получать следующие числовые коды: слово «ТАЛАКРАМА» – код 71, «АПАРАТ» – код 49.

В работе приводятся результаты сравнения алгоритма Polyphone с другими алгоритмами. В частности, процент правильно отождествленных слов у алгоритма Polyphone оценен как 95,12 %, в то время как наилучший результат после транслитерации русских слов по ГОСТ Р 52535.1-2006 у алгоритма NYSIIS составляет 75,97 %, у алгоритма SoundEx – 90,24 %, у алгоритма Metaphone – 90,29 %, у алгоритма Double Metaphone – 96,15 %, у алгоритма Daitch-Mokotoff Soundex – 96,84 %.

В то же время утверждается, что точность нечеткого сравнения слов достигает 98,8 %.

3. Фонетическое расстояние

Основным методом, реализуемым в рассмотренных ранее алгоритмах фонетического кодирования, является метод эквивалентных преобразований слова по звучанию, при котором часть слова, принадлежащая некоторому множеству (классу эквивалентности) заменяются кодом этого множества или его типичным представителем.

При этом заметно, что части слов из одного множества, близких по звучанию, также близки по написанию. При введении соответствующей метрики на словах можно поставить задачу определения схожести слов по звучанию путем подсчета расстояния между словами по написанию.

Этот подход используется в другом классе алгоритмов фонетического кодирования, в котором вычисление фонетического кода слова заменяется попарным сравнением слов путем вычисления расстояния между ними в некотором метрическом пространстве. Предполагается, что речевой и письменный строй естественного языка сильно коррелирует между собой. Осталось только найти соответствующую метрику.

3.1. РАССТОЯНИЕ ЛЕВЕНШТЕЙНА

Расстояние Левенштейна – это мера различия двух слов относительно минимального числа операций вставки, удаления и замены, необходимых для преобразования одного слова другое [2].

Расстояние Левенштейна $L(i, j)$ между двумя словами a и b длиной i и j при $\min(i, j) = 0$ по определению равно $\max(i, j)$, а при $\min(i, j) > 0$ находится из следующего рекуррентного уравнения:

$$L(i, j) = \min \begin{pmatrix} L(i, j-1) + 1, \\ L(i-1, j) + 1, \\ L(i-1, j-1) + m(i, j) \end{pmatrix},$$

где $m(i, j)$ равно нулю, если i-я буква слова a равна j-й букве слова b , и единице – в противном случае.

Установлено [18], что более 80 % ошибок написания слов составляют ошибки перестановки букв. Поэтому расстояние Левенштейна в настоящее время определяется не на трех, а на четырех операциях: вставка, удаление, замена и транспозиция.

Алгоритм вычисления расстояния Левенштейна $L(a, b)$ между словами a и b имеет вычислительную сложность $\Theta(|a|\cdot|b|)$[1] при объеме памяти $\Theta(\min(|a|, |b|))$, где $|x|$ – длина слова x [80].

Одной из проблем использования расстояния Левенштейна для определения фонетической близости двух слов является необходимость задания минимального расстояния между фонетически схожими словами ε. Чтобы минимизировать число ошибок используют функцию ε, зависящую от максимальной длины сравниваемых слов n. Обычно принимают $\varepsilon(n < 5) = 0$, $\varepsilon(4 < n < 9) = 1$, $\varepsilon(8 < n) = 3$.

Функции, вычисляющие расстояние Левенштейна, реализованы на многих языках программирования [11].

---

[1] *Знак* Θ *указывает на асимптотическое ограничение сверху и снизу,* $f(n) \in \Theta(g(n)) \leftrightarrow \exists(C, D, N > 0)\ \forall(n > N)\ |C\cdot g(n)| \le f(n) \le |D\cdot g(n)|$.

3.2. РАССТОЯНИЕ НА ОСНОВЕ N-ГРАММ

N-граммой называется последовательность из *N* элементов (букв). Для определения фонетической близости двух слов подсчитывается число общих *N*-грамм. Обычно *N* принимается равным трем.

Пример 7. Рассмотрим два слова, которые имеют одинаковое звучание: «Thomson» и «Thompson». Разобьём эти слова на триграммы. В результате получим, что слово «Thomson» включает триграммы «THO», «HOM», «OMS», «MSO» и «SON», а слово «Thompson» – «THO», «HOM», «OMP», «MPS», «PSO» и «SON». Общими триграммами этих слов являются триграммы «THO», «HOM» и «SON». Доля общих триграмм составляет 3/6, что определяет расстояние между словами, равное трем (подсчет велся относительно слова с большим числом триграмм).

Пример 8. Рассмотрим другие два слова: «Dane» и «Dean». Слово «Dane» имеет триграммы «DAN» и «ANE», слово «Dean» – «DEA» и «EAN». Хотя эти слова имеют одинаковое звучание, общие триграммы у них отсутствуют.

Как видно из примеров, различие между расстоянием, определенным с помощью триграмм, и расстоянием Левенштейна заключается в том, что различающаяся буква имеет значительное влияние на первое расстояние и слабое влияние на второе. Очевидно также, что вычислении расстояния между словами на основе *N*-грамм дает лучший результат для более длинным слов, чем для коротких.

Обычно *N*-граммы используются для нечеткого сравнения слов, которое не затрагивает фонетических аспектов. Например, для идентификации языка, так как установлено, что на достаточно длинных текстах каждый язык имеет свое распределение *N*-грамм, для сжатия текстов, для «угадывания» следующих букв и т.п. Другое применение *N*-грамм – индексирование данных в поисковых системах [33].

3.3. РАССТОЯНИЕ ДЖАРО

Неформальное определение расстояния Джаро между двумя словами – это минимальное число однобуквенных изменений,

которое необходимо выполнить для преобразования одного слова в другое [30, 31]. Чем меньше расстояние Джаро, тем более схожи сравниваемые слова.

Расстояние Джаро $D(a, b)$ между двумя словами a и b при $m > 0$ определяется так:

$$D(a,b) = w_1 \frac{m}{|a|} + w_2 \frac{m}{|b|} + w_3 \frac{m-t}{m},$$

где w_1, w_2, и w_3 – весовые коэффициенты, $w_1 + w_2 + w_3 = 1$; m – число совпадающих букв (число букв, разнесенных не более чем на половину длины самого короткого слова); t – половина числа транспозиций (половина числа совпадающих букв, отличающихся порядковыми номерами). При $m = 0$ по определению $D(a, b) = 0$.

Пример 9. Найдем расстояние Джаро между словами «Dane» и «Dean». Число совпадающих букв m равно четырем, а половина транспозиций t – 3/2. Приняв коэффициенты, равные 1/3, и округляя t до целого числа, получаем $d = 11/12$. После округления имеем $d = 1$. Действительно, слово «Dean» преобразуется в слово «Dane» путем транспозиции буквы E в конец слова.

В работе [81] предложено улучшение формулы вычисления расстояния Джаро исходя из предположения о том, что начало слова более значимо по сравнению с оставшейся его частью:

$$d'(a,b) = d(a,b) + s\left[1 - d(a,b)\right]/10,$$

где $d'(a, b)$ – улучшенное расстояние Джаро, s – число начальных общих букв слова, не превышающее четырех.

Следует заметить, что вычислительная сложность определения расстояния Джаро самая высокая из всех ранее рассмотренных алгоритмов [46]. Самая низкая вычислительную сложность оказалась у алгоритма вычисления расстояния на основе N-грамм.

4. Применение фонетического кодирования

Алгоритмы фонетического кодирования получили широкое распространение в современных информационных технологиях как в неизменном, так и в адаптированном виде, например:

- для нормализации данных в социальных сетях [9, 52, 68];
- для устранения дублирования данных [6, 49, 54, 57];
- для фонетического поиска [48, 79];
- при переводе с одного языка на другой [17];
- для поиска в медицинских базах данных [50, 74];
- для строковых метрик в распределенных базах данных [23];
- в поисковых сервисах сети интернет [3, 69];
- для извлечения семантических данных из онтологий [29];
- для отбора слов морфологических генераторов [37];
- для отождествления слов при восстановлении данных [82];
- для первичной обработки голосовых запросов [72];
- для распознавания чрезвычайных ситуаций [77];
- при анализе и синтезе речи [25, 61];
- при анализе данных в интеллектуальных системах [35];
- при обработке и хранении персональных данных [24, 39];
- при проверке правописания на различных языках [5, 43, 71];
- для создания хэш-тегов в технологии Big Data [13];
- для распознавания этнической принадлежности [0];
- при статистической идентификации объектов [19];
- для поиска речевых документов [66, 67];
- для генерации реалистичных персональных данных [15];
- для генерации хэш-ключей при анализе данных [16];
- при предсказании поведения пользователей [61];
- при интеграции данных на основе топонимов [73];
- для индексирования музыкальных произведений [26, 51];
- при обработке исторических документов [20];
- для исправления орфографических ошибок [32, 64];
- для ускоренного ввода текста с клавиатуры [55];
- для анализа настроения пользователей [76];
- для кросс-языкового извлечения данных [12];
- для определения сходства коротких сообщений [63];
- для межязыковой транслитерации [4, 8, 0];

– для распознавания слов при обработке текстов [0].

5. Заключение

Хотя основные алгоритмы фонетического кодирования предложены в прошлом веке, исследование и разработка других алгоритмов никогда не прекращались. Однако новых результатов в этой области получено не было, и основным итогом всех исследований явилось только улучшение базовых алгоритмов [34, 65].

Несмотря на все улучшения, общим недостатком алгоритмов фонетического кодирования все еще остается наличие в получаемых результатах ложноположительных и ложноотрицательных ошибок. Это объясняется тем, что эти алгоритмы работают не на последовательности элементарных звуков, из которых состоят слова, а на их текстовом представлении, которое, к тому же, искажено правилам письма того или иного естественного языка.

Очевидно, имеющиеся различия между языком-речью и языком-письмом делают алгоритмы фонетического кодирования не всегда корректными. Видится перспективным фонетическое преобразование сравниваемых слов в последовательность элементарных звуков (аллофонов, фонем, дифонов, трифонов) перед использованием базовых алгоритмов фонетического кодирования.

Теоретической основой такого подхода могут служить результаты, появившиеся в последнее время в области синтеза и распознавания речи. В частности, разработаны и опробованы алгоритмы, позволяющие как «озвучивать» текст, так и выделять из речевого потока последовательности элементарных звуков [33].

Литература

1. КАНЬКОВСКИ П. *«Как ваша фамилия?» или русский MetaPhone* //Программист. – 2002. – Вып. 8. – С. 36–39.
2. ЛЕВЕНШТЕЙН В.И. *Двоичные коды с исправлением выпадений, вставок и замещений символов* // Докл. АН СССР. – 1965. – Вып. 163 (4). – С. 845–848.
3. ABDULHAYOGLU M.A., THIJS B. *Use of ResearchGate and Google CSE for author name disambiguation* // Scientometrics. – Akademiai Kiado, Budapest, Hungary, 2017 – Vol. 111. –
   P. 1965–1985.
4. ADEEL Z.M., IQBAL R.N., MASOOD S.A. *English to Urdu transliteration: An application of Soundex algorithm* // IEEE Int. Conference on Information and Emerging Technologies, Karachi, Pakistan, June 14 –16, 2010. – P. 1–5.
5. ALMEIDA G., AVANCO L., DURAN M.S., et al. *Evaluating Phonetic Spellers for User-Generated Content in Brazilian Portuguese* // Int. Conference on Computational Processing of the Portuguese Language, Tomar, Portugal, June 13–15, 2016. – Springer International Publishing, 2016. – P. 361–373.
6. ANGELES M.P., PEREZ-FRANKO L.F. *Analysis of string encoding functions during de-duplication process* // Int. Conference on Informatics, Electronics & Vision, Fukuaka, Japan, June 15–18, 2015. – P. 1–6.
7. ANGELES M.P., ESPINO-GAMEZ A., GIL-MONCADA J. *Comparison of a Modified Spanish Phonetic, Soundex, and Phonex coding functions during data matching process* // Int. Conference on Informatics, Electronics & Vision, Fukuaka, Japan, June 15–18, 2015. – P. 1–5.
8. ANONTHANASAP O., KETNA M., LEELANUPAB T. *Automated English mnemonic keyword suggestion for learning Japanese vocabulary* // IEEE Int. Conference on Information Technology and Electrical Engineering (ICITEE), Chiang Mai, Thailand, October 29–30, 2015. – P. 638–643.
9. BILAL A. *Lexical normalization of Twitter Data* // Science and Information Conference, London, UK, July 28–30, 2015. – P. 326–328.

10. BINSTOCK A., REX J. *Practical Algorithms for Programmers*. – Addison-Wesley, 1995. – 577 p.
11. *Calculate Levenshtein distance between two strings*. – URL: http://php.net/manual/en/function.levenshtein.php (дата обращения: 08.09.2017).
12. CHHEDA P., FARUQUI M., MITRA P. *Handling OOV Words in Indian-language – English CLIR* // European Conference on Information, Barcelona, Spain, April 1–5, 2012. – Retrieval. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 476–479.
13. CHERICHI S., FAIZ R. *Upgrading Event and Pattern Detection to Big Data* // Int. Conference on Computational Collective Intelligence, Halkidiki, Greece, September 28–30, 2016. – Springer International Publishing, 2016. – P. 377–386.
14. CHOUDHURY M., SARAF R., JAIN V., MUKHERJEE A., SARKAR S, BASU A. *Investigation and modeling of the structure of texting language* // Int. Journal of Document Analysis and Recognition. – Springer-Verlag, 2007. – Vol. 10. – P. 157–174.
15. CHRISTEN P., PUDJIJONO A. *Accurate Synthetic Generation of Realistic Personal Information* // Pacific-Asia Conference on Knowledge Discovery and Data Mining, Bangkok, Thailand, April 27–30, 2009. – Berlin Heidelberg: Springer-Verlag, 2009. – P. 507–514.
16. CHRISTEN P. *Geocode Matching and Privacy Preservation* // Privacy, Security and Trust in KDD, Las Vegas, USA, August 24, 2008. – Berlin Heidelberg: Springer-Verlag, 2009. – P. 7–24.
17. CHUNG J.M., LU C.Y., LEE H.M., HO J.M. *Automatic English-Chinese name translation by using Web-Mining and phonetic similarity* // IEEE Int. Conference on Information Reuse & Integration. Las Vegas, USA, August 3–5, 2011. – P. 283–287.
18. DAMERAU F.J. *A technique for computer detection and correction of spelling errors* // ACM. – 1964. – No. 7(3). – P. 171–176.
19. DENK M. *Framework for Statistical Entity Identification in R* // Data Analysis, Machine Learning and Applications. – Berlin Heidelberg: Springer-Verlag, 2008. – P. 335–342.

20. DONGHUI L., DEWEI P. *Spelling Correction for Chinese Language Based on Pinyin-Soundex Algorithm* // Int. Conference on Internet Technology and Applications, Wuhan, China, August 16–18, 2011. – P. 1–3.
21. ERNST-GERLACH A., FUHR N. *Advanced Training Set Construction for Retrieval in Historic Documents* // Asia Information Retrieval Symposium, Taipei, Taiwan, December 1–3, 2010. – Berlin Heidelberg: Springer-Verlag, 2010. – P. 131–140.
22. GAONA M.A., GELBUKH A., BANDYOPADHYAY S. *Recognizing Textual Entailment Using a Machine Learning Approach* // Mexican Int. Conference on Artificial Intelligence, Pachuca, Mexico, November 8–13, 2010. – Berlin Heidelberg: Springer-Verlag, 2010. – P. 177–185.
23. GIRAUD-CARRIER C., GOODLIFFE J., JONES B.M. CUEVA S. *Effective record linkage for mining campaign contribution data* // *Knowledge and Information Systems* – London: Springer-Verlag, 2015. – Vol. 45. – P.389–416.
24. GRZEBALA P., CHEATHAM M. *Private Record Linkage: Comparison of Selected Techniques for Name Matching* // Int. Semantic Web Conference, Heraklion, Crete, Greece, May 29 – June 2, 2016. – Springer International Publishing, 2016. – P. 593–606.
25. JIAN H.-L. *Speech Driven by Artificial Larynx: Potential Advancement Using Synthetic Pitch Contours* // Int. Conference on Universal Access in Human-Computer Interaction, Los Angeles, USA, August 2–7, 2015. – Springer International Publishing, 2015. – P. 312–321.
26. HAN Y., MIN L., ZOU Y. et al. *LRC Sousou: A Lyrics Retrieval System* // Int. Conference of Young Computer Scientists, Engineers and Educators, Harbin, China, January 10–12, 2015. – Berlin Heidelberg: Springer-Verlag, 2015. – P. 464–467.
27. HERZOG T.N., SCHEUREN F.J., WINKLER W.E. *Data Quality and Record Linkage Techniques*. – New York: Springer, 2007. – P. 115-121.

28. HONG S.G., JANG S., CHUNG Y.H. et al. *News Media Analysis Using Focused Crawl and Natural Language Processing: Case of Lithuanian News Websites* // Int. Conference on Information and Software Technologies, Kaunas, Lithuania, September 13–14, 2012. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 48–61.
29. HU B., HU B. *On Capturing Semantics in Ontology Mapping* // LLC: Springer Science + Business 2008. – Vol. 11. – P. 361–385.
30. JARO M.A. *UNIMATCH – a computer system for generalized record linkage under conditions of uncertainty* // Spring Joint Computer Conference, Anaheim, USA December 5–17, 1972. – P. 523–530.
31. JARO M.A. *Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida* // Journal of the American Statistical Association. – 1989. –No. 84(406). – P. 414–420.
32. JORDAO C.C., ROSA J.L. *Metaphone-pt_BR: The Phonetic Importance on Search and Correction of Textual Information* // Int. Conference on Intelligent Text Processing and Computational Linguistics, New Delhi, India, March 11–17, 2012. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 297–305.
33. JURAFSKY D., MARTIN J.H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. – Pearson Prentice Hall, 2009. – 988 p.
34. KARAKASIDIS A., VERYKIOS V.S. *Privacy Preserving Record Linkage Using Phonetic Codes* // Balkan Conference in Informatics, Thessaloniki, Greece, September 17– 19, 2009. – P. 101–106.
35. KARAKASIDIS A., KOLONIARI G., VERYKIOS V.S. *Privacy Preserving Blocking and Meta-Blocking* // Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Porto, Portugal, September 7–11, 2015. – Springer International Publishing, 2015. – P. 232–236.

36. KAVEH-YAZDY F., ZAREH-BIDOKI A.M. *Aleph or Aleph-Maddah, that is the question! Spelling correction for search engine autocomplete service* // Int. Conference on Computer and Knowledge, Mashhad, Iran, October 29–30, 2014. – P. 273–278.
37. KNUTH D.E. *The Art of Computer Programming*. *Vol. 3*. / Second Edition. – Addison-Wesley, 1998.
38. KOCOŃ J., PIASECKI M. *Named Entity Matching Method Based on the Context-Free Morphological Generator* // Int. Conference on Natural Language Processing, Warsaw, Poland, September 17–19, 2014. – Springer International Publishing, 2014. – P. 34–44.
39. KROLL M., STEINMETZER S. *Who Is 1011011111... 1110110010? Automated Cryptanalysis of Bloom Filter Encryptions of Databases with Several Personal Identifiers* // Int. Joint Conference on Biomedical Engineering Systems and Technologies, Lisbon, Portugal, January 12–15, 2015. – Springer International Publishing, 2015 – P. 341–356.
40. LAWRENCE P. *Hanging on the Metaphone* // Computer Language. – 1990. – Vol. 7, No. 12. – P. 39–44.
41. LAWRENCE P. *The Double Metaphone Search Algorithm* // C/C++ Users Journal. – 2000. – No. 18(6). – P. 38–43.
42. LAWRENCE P. *Metaphone 3: Version 2.1.3*. – URL: https://searchcode.com/codesearch/view/2366000/ (дата обращения: 18.02.2012).
43. LI J., OUAZZANE K., JING Y. et al. *Evolutionary Ranking on Multiple Word Correction Algorithms Using Neural Network Approach* // Int. Conference on engineering Applications of Neural Networks, London, UK, August 27–29, 2009. – Berlin Heidelberg: Springer-Verlag, 2009. – P. 409–418.
44. LI L., QUAZZANE K., KAZEMIAN H., JING Y., BOYD R. *A neural network based solution for automatic typing errors correction* // Neural Computing and Applications. – 2011. – Vol. 20. – P. 889–896.
45. LISBACH B., MEYER M. *Linguistic Identity Matching*. – Springer Fachmedien Wiesbaden, 2013. – P. 118–120.

46. LOUPPE G., AL-NATSHEH H.T., SUSIK M., MAGUIRE E.J. *Ethnicity Sensitive Author Disambiguation Using Semi-Supervised Learning* // Int. Conference on Knowledge Engineering and the Semantic Web, Prague, Czech Republic, September 21–23, 2016. – Springer International Publishing, 2016. – P. 272–287.
47. MAARIF H.A., AKMELIAWATI R., HTIKE Z.Z., GUNAWAN T.S. *Complexity Algorithm Analysis for Edit Distance* // Int. Conference on Computer and Communication Engineering, Kuala Lumpur, Malaysia September 23–24, 2014. – P. 135–137.
48. MANDAL A.K., HOSSAN D., NADIM. *Developing an efficient search suggestion generator, ignoring spelling error for high speed data retrieval using Double Metaphone Algorithm* // Int. Conference on Computer and Information Technology, Dhaka, Bangladesh, December 23–25, 2010. – P. 317–320.
49. MARTINS B. *A Supervised Machine Learning Approach for Duplicate Detection over Gazetteer Records* // Int. Conference on GeoSpatial Semantics, Brest, France, May 12–13, 2011. – Berlin Heidelberg: Springer-Verlag, 2011. – P. 34–51.
50. MASON-BLAKLEY F., LU L., PRICE M., ROUDSARI A. *An RCT Simulation Study on Performance and Accuracy of Inexact Matching Algorithms for Patient Identity in Ambulatory Care Settings* // Int. Conference on Healthcare Informatics, Dallas, USA, October 21–23, 2015. – P. 8–17.
51. MEINTANIS K., SHIPMAN F.M. *Visual Expression for Organizing and Accessing Music Collections in MusicWiz* // Int. Conference on Theory and Practice of Digital Libraries, Glasgow, UK, September 6–10, 2010. – Berlin Heidelberg: Springer-Verlag, 2010. – P. 80–91.
52. MOSQUERA A., MOREDA P. *The Study of Informality as a Framework for Evaluating the Normalisation of Web 2.0 Texts* // Int. Conference on Application of Natural Language to Information Systems, Groningen, The Netherlands, June 26–28, 2012. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 241–246.

53. MUTALIB A., NOAH S.A. *Phonetic coding methods for Malay names retrieval* // IEEE Int. Conference on Semantic Technology and Information, Putrajaya, Malaysia June 28–29, 2011. – P. 125–129.
54. MUTHMANN K., LOSER A. *Detecting Near-Duplicate Relations in User Generated Forum Content* // OTM Confederated International Conferences «On the Move to Meaningful Internet Systems», Hersonissos, Crete, Greece, October 25–29, 2010. – Berlin Heidelberg: Springer-Verlag, 2010. – P. 698–707.
55. OUAZZANE K., LI J., KAZEMIAN H.B. *An Intelligent Keyboard Framework for Improving Disabled People Computer Accessibility* // Engineering Applications of Neural Networks, Corfu, Greece, September 15–18, 2011. – International Federation for Information Processing, 2011. – P. 382–391.
56. OUSIDHOUM N.D., BENSAOU N. *Towards the Refinement of the Arabic Soundex* // Int. Conference on Application of Natural Language to Information Systems, Salford, UK, June 19–21, 2013. – Berlin Heidelberg: Springer-Verlag, 2013. – P. 309–314.
57. OWONIBI M., KOENIG-RIES B. *A Quality Management Workflow Proposal for a Biodiversity Data Repository* // Int. Conference on Conceptual Modeling, Atlanta, USA, October 27–29, 2014. – Springer International Publishing, 2014. – P. 157–167.
58. PANDE B.P., DHAMI H.S. *Application of Natural Language Processing Tools in Stemming* // Int. Journal of Computer Applications. – 2011. – No. 27(6). – P. 14–19.
59. PARAMONOV V.V., SHIGAROV A.O., RUZHNIKOV G.M. et al. *Polyphon: An Algorithm for Phonetic String Matching in Russian Language* // Int. Conference on Information and Software Technologies, Druskininkai, Lithuania, October 13–15, 2016. – Springer International Publishing, 2016. – P. 568–579.

60. PARMAR V.P., PANDYA A.K., KUMBHARANA C.K. *Determining the character replacement rules and implementing them for phonetic identification of given words to identify similar pronunciation words* // IEEE Int. Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE), New Delhi, India, February 25–27 , 2015. – P. 272–277.
61. PARVATHY A.G, VASUDEVAN G.B., KUMAR A., BALAKRISHNAN R. *Leveraging Call Center Logs for Customer Behavior Prediction* // Int. Symposium on Intelligent Data Analysis, New Delhi, India, February 25–27, 2015. – Berlin Heidelberg: Springer-Verlag, 2009. – P. 143–154.
62. PFEIFFER A., KURAEVA A., FOULONNEAU M. et al. *Automatically Generated Metrics for Multilingual Inline Choice Questions on Reading Comprehension* // Int. Computer Assisted Assessment Conference, Zeist, The Netherlands, June 22–23, 2015. – Springer International Publishing, 2015. – P. 80–95.
63. PINTO D., VILARINO D., ALEMAN Y. et al. *The Soundex Phonetic Algorithm Revisited for SMS Text Representation* // Int. Conference on Text, Speech and Dialogue, Brno, Czech Republic, September 3–7, 2012. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 47–55.
64. PURAO S., STOREY V.C., SUGUMARAN V. et al. *Repurposing Social Tagging Data for Extraction of Domain-Level Conc*epts // Int. Conference on Application of Natural Language to Information Systems, Alicante, Spain, June 28–30, 2011. – Berlin Heidelberg: Springer-Verlag, 2011. – P. 185–192.
65. RAKIBUL H., REAZ A. *FM-Chord: Fault-tolerant Chord supporting misspelled queries* // Int. Conference on Computers and Information Technology, Dhaka, Bangladesh, December 21–23, 2009. – P. 651–656.
66. REYES-BARRAGAN M., VILLASENOR-PINEDA L., MONTES-Y-GOMEZ M. *A Soundex-Based Approach for Spoken Document Retrieval* // Mexican Int. Conference on Artificial Intelligence, Atizapan de Zaragoza, Mexico, October 27–31, 2008. – Berlin Heidelberg: Springer-Verlag, 2008. – P. 204–211.

67. REYES-BARRAGAN A., MONTES-Y-GOMEZ M., VILLASENOR-PINEDA L. *Combining Word and Phonetic-Code Representations for Spoken Document Retrieval* // Int. Conference on Intelligent Text Processing and Computational Linguistics, Tokyo, Japan, February 20–26, 2011. – Berlin Heidelberg: Springer-Verlag, 2011. – P. 458–466.
68. ROEDLER R., KERGL D., RODOSEK G.D. *Profile Matching Across Online Social Networks Based on Geo-Tags* // Advances in Nature and Biologically Inspired Computing. – Springer International Publishing, 2016. – P. 417–428.
69. RUSSELL R.C, MARGARET K.O. *US Patent 1262167, 1435663.* – 1918, 1922.
70. SANCHEZ-VILAS F., LAMA M., VIDAL J.C. et al. *Combining Uncorrelated Similarity Measures for Service Discovery* // Int. Workshop on Resource Discovery, Paris, France, November 5, 2010. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 160–180.
71. SCHIERLE M., SCHULZ S., ACKERMANN M. *From Spelling Correction to Text Cleaning – Using Context Information* // Data Analysis, Machine Learning and Applications. – Berlin Heidelberg: Springer-Verlag. – 2008. – P. 397–404
72. SCHNEIDER J.M., FERNANDEZ J., MARTINEZ P. *A Proof-of-Concept for Orthographic Named Entity Correction in Spanish Voice Queries* // Int. Workshop on Adaptive Multimedia Retrieval, Copenhagen, Denmark, October 24–25, 2012. – Springer International Publishing, 2014. – P. 181–190.
73. SMART P.D., JONES C.B., TWAROCH F.A. *Multi-source Toponymal Data Integration and Mediation for a Meta-Gazetteer Service* // Int. Conference on Geographic Information Science. Zurich, Switzerland, April 30, 2010. – Berlin Heidelberg: Springer-Verlag, 2010. – P. 234–248.
74. SOMAN S., SRIVASTAVA P, MURTHY B.K. *Unique Health Identifier for India: An algorithm an feasibility analysis on patient data* // Int. Conference on E-health Networking, Application & Services. Boston, USA, October 13–17, 2015. – P. 250–255.

75. *Soundexing and Genealogy by Gary Mokotoff*. – URL: http://www.avotaynu.com/soundex.htm (дата обращения: 08.09.2017).
76. SOUZA M., VIEIRA R. *Sentiment Analysis on Twitter Data for Portuguese Language* // Sentiment Analysis on Twitter Data for Portuguese Language. – Berlin Heidelberg: Springer-Verlag, 2012. – P. 241–247.
77. TAFT R.L. *Name Search Techniques* // New York State Identification and Intelligence system: Special Report No. 1. – New York: Albany, 1970.
78. SOUZA J., BOTEGA L.C., SEGUNDO S. et al. *Conceptual Framework to Enrich Situation Awareness of Emergency Dispatchers* // Int. Conference on Human Interface and the Management of Information, Los Angeles, USA, August 2–7, 2015. – Springer International Publishing, 2015. – P. 33–44.
79. TISSOT H., PESCHL G., FABRO M.D. *Fast Phonetic Similarity Search over Large Repositories* // Int. Conference on Database and Expert Systems Applications. Munich, Germany, September 1–4, 2014. – Springer International Publishing, 2014. – P. 74–81.
80. WAGNER R.A., FISCHER M.J. *The string-to-string correction problem* // ACM. – 1974. – No. 21(1). – P. 168–173.
81. WINKLER W.E. *String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record* // Proc. of the Section on Survey Research Methods. – American Statistical Association, 1990. – P. 354–359.
82. ZAMPIERI M., AMORIM R.C. *Between Sound and Spelling: Combining Phonetics and Clustering Algorithms to Improve Target Word Recovery* // Int. Conference on Natural Language Processing, Warsaw, Poland, September 17–19, 2014. – Springer International Publishing, 2014. – P. 438–449.

OVERVIEW OF PHONETIC ENCODING ALGORITHMS

Valeriy Vykhovanets, Trapeznikov Institute of Control Sciences of RAS, Moscow, Dr. Sc., associative professor (valery@vykhovanets.ru).
Jianming Du, Bauman Moscow State University, Moscow, postgraduate student (forsola@qq.com)
Sergey Sakulin, Bauman Moscow State University, Moscow, PhD, associative professor (ss141291@yandex.ru)

Abstract: This paper gives an overview of the phonetic encoding algorithms, designed to determine the similarity of words in sound (pronunciation). Phonetic encoding algorithms are divided into algorithms for comparing words and algorithms for determining the distance between words. Word comparison algorithms such as SoundEx, NYSIIS, Daitch-Mokotoff, Metaphone, Polyphone and algorithms for determining the distance between words such as Levenshtein, Jaro, N-grams are described. For each algorithm, its advantages and disadvantages are indicated, an analogue of the algorithm for the Russian language is given. To eliminate the common shortcomings of phonetic encoding algorithms, it is proposed to use not the sequence of letters of words, but the sequence of their elementary sounds. In this case, word recognition, record linkage, indexing words by sounds are expected to improve.

Keywords: phonetic encoding algorithms, phonetic distance, record linkage, indexing words by sound.

Статья представлена к публикации членом редакционной коллегии М.Ф. Караваем.

Поступила в редакцию 12.09.2017.
Опубликована 31.05.2018.