# Analysis and Comparative Study on Phonetic Matching Techniques

Rima Shah
Student (ME CSE IV)
Parul Institute of Engineering and Technology

Dheeraj Kumar Singh
Assistant Professor
Parul Institute of Engineering and Technology

## ABSTRACT
Searching for names in large databases containing spelling variations has always been a problem. The first solution to the problem was proposed by Robert Russell in 1912 as he proposed SoundEx algorithm. SoundEx algorithm matches the names based on the sound of the words. The technique which is based on the pronunciation of the word is known as phonetic matching. SoundEx algorithm is one of the phonetic matching algorithms. We have discussed the other phonetic matching algorithm like edit distance algorithm, K-String and Q gram algorithm, Guth algorithm, Daitch Mokotoff algorithm and Metaphone coding algorithm. Our main focus is on the SoundEx algorithm and this paper also describes the issues with SoundEx algorithm.

## Keywords
Phonetic matching, SoundEx algorithm, Name variations

## 1. INTRODUCTION
Phonetic comparison algorithms are precisely defined methods for quantifying the similarity between speech forms or segments, words, or even entire languages on the basis of their sounds[5].Phonetic matching is used to identify the strings that sounds similar, meaning that the string having the similar pronunciations regardless of their spelling. Phonetic matching is used to evaluate the similarity of pronunciations of pairs of strings [10]. It focuses on the pronunciation of the word independent the spelling of words. For this many phonetic matching algorithms are used like Edit distance algorithm, SoundEx algorithm, string and /Q gram algorithm, etc.. The most common issue with name matching is Name Variations. The sources of name variations are as follows:

(1) Spelling variations. These types of variations include misplaced letters due to typographical errors, substituted letters, additional letters, or omissions [4]. Generally, such variations do not affect the phonetic structure of the name but still cause problems with matching algorithms.

(2) Phonetic variations. Where the phonemes of the name are modified, e.g. through mishearing [4], the structure of
The name is significantly changed. Sinclair and St. Clair are related names, but their phonetic structure is very different.

(3) Double names. In some cases it may be possible that the surname is composed of two elements [4]. But may not be necessary shown both at one time. E.g. double surname Vaughan Williams can be used as either Vaughan or Williams instead of full surname Vaughan Williams.

(4) Double first names. Although not common in the English language, when considering French, for example, names such as Jean-Claude may be given in full, or as Jean and/or Claude [4].

(5) Where an individual changes their name during the course of their life or is called by one of their first names during a period of their life and another later on, this causes a major problem in the name-matching. In these situations an algorithm that recognizes simple variations in spelling or phonetics would not be able to identify two such names as referring to the same person [4].

Searching names in large database have always been a problem. The solution to the problem was given by Robert Russell in1912 as he proposed SoundEx algorithm [11].

The names might be misspelled in a large database or might not be spelled as one expected. In this case rather than looking for exact matching, searching for approximate matching will be significant [7, 8].

One solution is to say that two names are approximate matches if they sound the same. And a method known as SoundEx was developed to determine if two names sound similar [9]. SoundEx is a system whereby values are assigned to names in such a manner that similar-sounding names get the same value. These values are known as SoundEx encodings. A search application based on SoundEx will not search for a name directly but rather will search for the SoundEx encoding.Based on the SoundEx encoding the similar sounding names would be retrieved from the large database.

## 2. PHONETIC MATCHING
Phonetic Matching performs the matching operation based on the pronunciation of words[10].To understand the working of matching operation we will discuss the example of large database that consists of the names Stefan, Steph, Stephen, Steve, Steven, Stove, and Stuffin. Suppose that we want to search for the name Stephen.

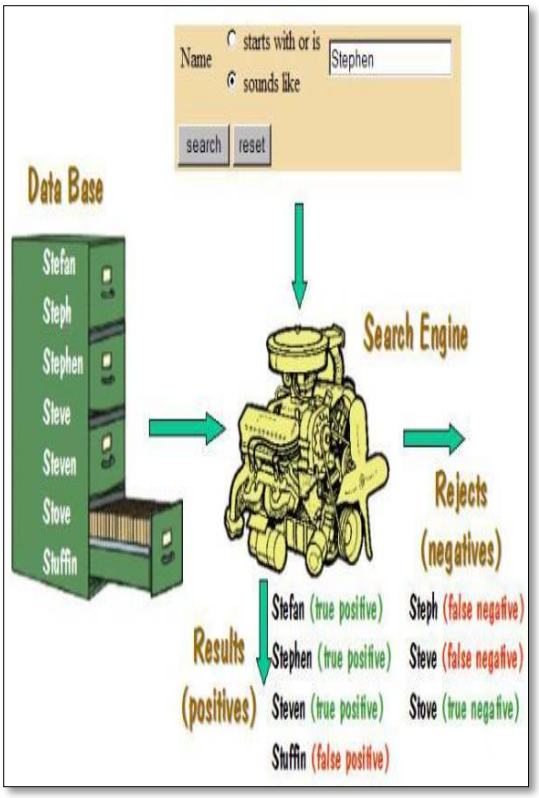


**Figure.1 True and False Negative [2].**

The matches that the search finds are called the positives, and those names that it rejects are called the negatives. Those positives that are relevant are called truepositives, and the others are false positives [2].As an example, let us assume that the matches found when searching for Stephen in the above database are Stefan, Stephen, Steven, and Stuffin. The first three are probably relevant, and are names that we would have wanted to see. So these are the true positives. Stuffin, however, is probably not relevant – it is a false positive.

The names that were rejected are Steph, Steve, and Stove. Of those, Stove is probably not one that we would have wanted. So it is a true negative. But Steph and Steve are ones that we would probably be interested in. They are false negatives [2].

# 3. PHONETIC MATCHING TECHNIQUES

A large number of researches are already being carried out in a well known area of Information retrieval under data mining. One of such technique of information retrieval is Phonetic matching which is used to compare the name based on the pronunciation of the words. The similar sounding words could be retrieved from the large database. For this, many name matching algorithms are used like SoundEx algorithm, Edit Distance algorithm, K-String and Q gram algorithm, Guth algorithm, Daitch Mokotoff algorithm, Metaphone coding algorithm.

## 3.1 Soundex Algorithm

SoundEx is the best-known phonetic matching scheme. Developed by Odell and Russell, and patented in 1918, SoundEx uses codes based on the sound of each letter to translate a string into a canonical form of at most four characters, preserving the first letter [1].

1. Retain the first letter of the string
2.Change all occurrences of the following letters to zero: a, e, h, i, o, u, w, y
3. Assign numbers to the remaining letters (after the first) as follows:
   b, f, p, v = 1
   c, g, j, k, q, s, x, z = 2
   d, t = 3
   l = 4
   m, n = 5
   r = 6
4. Remove all pairs of digits which occur beside each other from the string that resulted after the previous step.
5. Remove all the zeros from string that results from the previous step.
6. Return the first four characters, right-padding with zeroes if there are fewer than four.

**Figure.2: Outline of SoundEx Algorithm [1].**

Taking an example we will see how SoundEx algorithm works.
Example-"SMITH" will code to "25030" which will then reduce to "2530"by computing the steps of SoundEx algorithm.

### *3.1.1 Issues of Soundex Algorithm*

**a. Dependence on initial letter.**
The SoundEx algorithm uses initial letter as key component. As a result name with different starting letter will never match each other [6]. E.g. A data entry operator hearing the name "Corth" might type in the much more common "Korth" Although "Corth" may be in the database, it will not be returned by a standard SoundEx search on "Korth."

**b. Noise intolerance.**
SoundEx cannot search transposition error [6]. E.g. .If a database record contains the name "Hreman", SoundEx cannot overcome this simple transposition when searching for the correctly spelled "Herman".

**c. Differing transcription systems**.
The SoundEx codes for the members of these spelling variants do not always match each other, so one form of the name will not reliably retrieve the others [6]. The same Russian surname may occur as "Ivanov" or "Ivanoff" or even as "Iwanow". But because of this issue the code may be different for all of three names and hence cannot be retrieved for each other from database.

**d. Names containing particles**.
Names in many cultures contain optional or supplemental elements that may be present in one instance of a name, but missing from the next [6].The Arabic name "MohamedAli," for example, can also appear without the particle "Mohamed" as "Ali". Both of these variants can refer to the same person. SoundEx algorithm does not consider them as variant of same person's name and codes them as different names.

**e. Silent consonants**.
SoundEx is not able to capture the phonetic similarity between names with silent consonants [6]. E.g. an uncommon name like "Meghburn" may be spelled as "Meburn,". SoundEx assigns different codes to these pairs, such that they can never match each other. As a result these two names will never be retrieved for each other.

**f. Name syntax variation**.
Differing name structures are used by various cultures and societies. The most common structure is "first-middle-last" structure used with many different cultures around the world. As a result, names may be mapped inconsistently into the fields of database records.[3] The name "Sheikh Asif Mohamed" might be found in one database record with "Sheikh" as the first name, "Asif" as the middle name, and "Mohamed" as the last name. In another record, "Sheikh Asif" might be entered as the first name and "Mohamed" as the last name. SoundEx was not designed to deal with this type of variation in the form of a name.

**g. Initials**.
Initials are often substituted for full names [6]. Records for a "Mikhail Kovalchuk" and "M.Kovalchuk," for example, may well belong to the same person. However, a standard SoundEx query on "Mikhail Kovalchuk" will not retrieve "M. Kovalchuk" and vice versa.

**h. Poor precision**.
The SoundEx algorithm reduces distinctions between strings of letters to such a degree that many obviously dissimilar names are typically returned for each search transaction[6].For example, "Courtmanche," "Corradino," "Cartmill," and "Cortinez" were returned on a query for "Criton." This superfluous information has very real costs for application designers, in terms of processing resources consumed, response times and user satisfaction. Worse yet, the precision degrade as the database size increases, for many typical SoundEx applications.

### *3.1.2. Uses and Limitations of Soundex Algorithm*
SoundEx algorithm is basically used in name retrieval. To be more specific we can say that it can be used in application like Airline reservation. The SoundEx code for a passenger's surname is often recorded to avoid the confusion when trying to pronounce it. The SoundEx algorithm can also be used to avoid the problem when dealing with names that might have alternate spellings.

SoundEx acts as a bridge between the fuzzy and inaccurate process of human verbal interaction, and the concise true/false processes at the foundation of computer communication. As such, SoundEx is an inherently unreliable interface.

For this reason, SoundEx is only usable in applications that can tolerate high false positives and high false negatives.

## 3.2 K-String and Q-Gram Algorithm
K-Strings and Q-Grams are comparison algorithm that compare words or text and determine the closeness within those strings. As these algorithms finds discrepancy between two strings, they can be used when both strings are known. The most basic of this family of algorithms is K-String and Q-gram. A K-String algorithm searches for a specific string of text within a larger section of text, and counts the number of letter differences between the two [3]. K- String algorithm can be used in to compare first names and surnames [12]. Guth enhances this by taking into account the actual positions of letters. Obviously this technique is simple and not especially effective, but is often used for simple spell-checking algorithms. The advantage of these algorithms is that they are not tied to any phonetics of any specific language.

## 3.3 Edit Distance Algorithm
Edit distances are measures of string comparison. A simple edit distance algorithm, which counts the least number of single-character insertions, deletions, and replacements needed to transform one string into another. Edit Distance could be used for phonetic matching since similar-sounding words are often spelled similarly [1].
For example, the Levenshtein distance between "bitten" and "sitting" is 3, since the following three edits change one into the other, and there is no way to do it with fewer than three edits:

1. bitten → sitten (substitution of "s" for "b")
2. sitten → sittin (substitution of "i" for "e")
3. sittin → sitting (insertion of "g" at the end).

## 3.4 Guth Algorithm
Guth counts the number of differences between strings and the number of individual letter operations (insertions, deletions, and substitutions) required to achieve a match [3]. A score is returned as 0.0 to 1.0. In this case 0.0 indicates no match and 1.0 indicates exacts match. Guth algorithm is easy to code. The method is independent of language issues [12]. But do not work well when used for short names.

## 3.5 Daitch-Mokotoff
Daitch Mokotoff is Phonetic matching algorithm which was designed for greater accuracy and used for matching the Slavic and Yiddish surnames having similar pronunciations but different spellings [3]. In Daitch Mokotoff algorithm the names are coded up to six digits. Unlike the SoundEx algorithm in this algorithm first letter of the name is also coded. Traditional SoundEx algorithm returns only single encoding for single name. While Daitch Mokotoff algorithm returns multiple possible encoding for single name.

## 3.6 Metaphone Coding
Metaphone coding algorithm is developed by Lawrence Phillips replace the words by their phonetic equivalence [3]. It uses the set of English pronunciations rules.
1. Ignore vowels after the first letter and reduces the remaining alphabet to sixteen consonant sounds, but vowels are retained when they are the first letter.
2. Duplicate letters are not taken into account to the code.
3. Zero is used to represent the 'th' sound and 'X' is used for the 'sh' sound.
4. The sixteen consonant sounds are: B X S K J T F H L M N P R Ø W Y
Metaphone coding algorithm is not strictly name matching algorithm, but can be applied to compare both first names and surnames. This algorithm works well for English language [12].

# 4. COMPARISON
The table depicts the comparison of all the phonetic matching algorithms based on language specification and the features and limitations.

**Table 4.1 Comparison Table of Phonetic Matching Techniques**

| Algorithm | Language Specification | Features and Limitations |
|---|---|---|
| SoundEx Algorithm | Mostly Used for US English | Easy to implement and usable in applications that can tolerate high false positives and high false negatives. |
| K-String and Q gram Algorithm | Not tied to any phonetics of any specific language. | Simple but not effective, Can be used as Spell checking algorithm. |
| Edit distance Algorithm | N/A | Counts the least number of operations needed to transform one string into another. |
| Guth Algorithm | Independent of Language issue. | Do not work well with short names. |
| Daitch - Mokotoff | Designed to provide greater accuracy when used for Yiddish and Salvic surnames. | Use Six digits to code and returns multiple encoding for single name. |
| Metaphone Coding | Works well for English language. | Not strictly name matching algorithm, Used to compare both first name and surname. |

## 5. CONCLUSION

The phonetic matching is very important and complex technique, but needed in every part of the life. Different Phonetic Matching algorithms can be used for this purpose as per the requirement and language specifications. Different name matching algorithms have their own merits and demerits. By applying the most suitable techniques, we can increase the accuracy and efficiency of the system. Future research will focus on using these algorithms together, such that the strengths and efficiency of these techniques can be increased.

## 6. REFERENCES

[1] Hettiarachchi, G.P., Attygalle, D., "SPARCL: An improved approach for matching Sinhalese words and names in record clustering and linkage", IEEE, Colombo, 2012.

[2] Beider, A, Stephen P. Morse, Phonetic Matching: A Better Soundex, March, 2010.

[3] Chakkrit Snae, A Comparison and Analysis of Name Matching Algorithms, World Academy of Engineering and Technology 1, 2007.

[4] Name and Address Matching Strategy, White Paper, 2007 December.

[5] Brett Kessler, Phonetic Comparison Algorithms, Transaction of Philological Society Volume 103:2 243-260, 2005.

[6] Frankie Patman LeonardShaefer, Is Soundex Good Enough for You? On the Hidden Risks of SoundEx-Based Name Searching, Language Analysis Systems, Inc. 2214 Rock Hill Road, Suite 201 Herndon, VA 20170, 2001-2003

[7] Hall, P. A. V., and Dowling, G. R. (1980), Approximate String Comparison, Computing Surveys, 12, 381-402.

[8] P.A.V. Hall, G.R. Dowling, Approximate string matching. Computing Surveys, 12(4):381{402, 1980.

[9] Peter Christian, SoundEx - can it be improved? March, 1998.

[10] Justin Zobel, Philip Dart, Phonetic String Matching: Lessons from Information Retrieval.

[11] Beider, A, Stephen P. Morse, PhoneticMatching: An Alternative to SoundEx with Fewer False Hits.

[12] A.J.Lalit, B Randell, An assessment of name matching algorithms, Department of Computing Science University of Newcastle uponTyne.