Implement Phonetic ("Sounds-like") Name Searches with Double Metaphone Part VI: Other Methods & Additional Resources

By Adam Nelson.

Surveys other phonetic matching techniques, and presents additional resources on the subject.

Abstract

Simple information searches -- name lookups, word searches, etc. -- are often implemented with a match criterion. However, given both the diversity of homophonic (pronounced the same word, but possibly spelled differently) names as well as the propensity for humans to misspell surnames, this simplistic criterion often yields less accurate results, in the form of reduced result sets, missing records that differ by a misplaced letter spelling.

This article series discusses Lawrence Phillips’ Double Metaphone phonetic matching algorithm and several useful implementations which can be employed in a variety of solutions to create effective searches of proper names in databases and other collections.

Introduction

This article series discusses the practical use of the Double Metaphone algorithm to phonetic search using the author’s implementations written for C++, COM (Visual Basic, etc.), scripting (ASP), SQL, and .NET (C#, VB.NET, and any other .NET language). For a discussion of the algorithm itself, and Phillips’ original code, see Phillips’ article in the June 2000 CUJ, available at CodeProject.com.

Part I introduces Double Metaphone and describes the author’s C++ implementation and the use of the author’s COM implementation from within Visual Basic. Part III demonstrates implementation from ASP and with VBScript. Part IV shows how to perform phonetic matching using the author’s extended stored procedure. Part V demonstrates the author’s .NET implementation with VI closes with a survey of phonetic matching alternatives, and pointers to other resources.

Double Metaphone limitations

While this article series has focused entirely on the Double Metaphone algorithm as a method for phonetic matching, it is important to note that this algorithm has certain limitations and may not be suitable for all applications. For instance, it may not perform well for names that are not homophonic, or for names that have been misspelled in a way that is not captured by the algorithm. Additionally, the algorithm may not be able to handle names that are misspelled in multiple ways, or that are spelled differently in different parts of the world.
phonetic matching in one’s applications, Double Metaphone bears some weaknesses that for a particular application, including:

- Though it works as a general-purpose phonetic search algorithm, Double Metaphone works best with, searching lists of proper names rather than large fields of generic
- Double Metaphone provides minimal ranking ability, apart from the three match level in the series. This limits the ability to tune search results.
- Being a phonetic matching (vs. fuzzy matching like q-grams and edit distances) algorithm, Double Metaphone may fail to match misspelled words when the misspelling substantively structure of a word.

Even bearing these limitations in mind, Double Metaphone is free, efficient, easy to use, number of scenarios. Ultimately, only the designer of a particular system can decide if Double Metaphone is appropriate to his/her particular problem space.

Alternatives to Double Metaphone

Numerous other algorithms and techniques have been developed, each for different pur varying efficacy. This section will explore some of the better-known techniques, and provide information on each method.

Soundex

Soundex was one of the first, if not the first, formalized phonetic matching algorithm. So use by the US Census in the late 19th century. Not surprisingly, this algorithm is remark inadequate in most cases.

Nonetheless, one encounters Soundex in surprising places, even in modern software solu Microsoft SQL Server offers a `SOUNDEX` function which, given a word, computes Soundex.

For more information on Soundex, a simple Internet search on "soundex" will likely yield again, the reader is encouraged to consider more advanced alternatives for any production both primitive and limited.

Metaphone

Double Metaphone is only the latest incarnation of the Metaphone algorithm, originally pu Phillips in 1990. While arguably inferior to Double Metaphone, Metaphone does incorpor has the added advantage (and disadvantage) of producing only one phonetic key for a gi

Phonix

Phonix is an improved version of Soundex, developed by T.N. Gadd and published in Asss Management’s journal, Program [Gadd, T.N. ‘Fishing for words’: phonetic retrieval of systems, 22(3) 1988, p. 222] and [Gadd, T.N. PHONIX: the algorithm, 24(4) 1990, p. 36]. Not available online, Phonix has been incorporated into a number of WAIS implementation which is open-source and therefore freely available in source form.

The author has never experimented with Phonix, and therefore cannot write authoritative performance; however being Soundex-based, it bears much of the same baggage which performance. The paper by Zobel and Dart referenced at the end of this article performs comparison of Phonix with several other algorithms, producing results which confirm this
q-Gram based algorithms

A q-gram (sometimes called n-gram, primarily to confuse readers) in this context refers to letters long, from a given word. For example, for \( q = 2 \), the word Nelson has the following q-grams:

NE EL LS SO ON

By comparison, Neilsen breaks down into these q-grams (\( q = 2 \)):

NE EI IL LS SE EN

Clearly, Nelson and Neilsen share the NE and LS q-grams in common.

Various techniques have been developed which compare two words based on their q-grams. One would be counting the number of q-grams two words have in common, with a higher count indicating a better match.

Technically, q-gram algorithms aren’t strictly phonetic matching, in that they do not operate on the phonetic characteristics of words. Instead, q-grams can be thought to compute the difference, between two words. Since phonetically similar words often have similar spellings, q-grams can provide favorable results, yet it also successfully matches misspelled or otherwise mutated words rendered phonetically disparate.

Edit distance based algorithms

Edit distance computes the "distance" between two words by counting the number of insert, replace, and delete operations required to permute one word into another. In general the fewer operations required to match two words, the more similar they are. Some implementations assign varying scores to the insert, replace, and delete operations, and common variation varies which operations are considered when computing the distance; operation may not be considered, thereby defining the edit distance solely in terms of inserts.

One of the more popular algorithms in the edit distance class is Levenshtein distance (no references use the spelling 'Levenstein', which is technically incorrect).

As with q-gram algorithms, edit distance is not strictly phonetic, but often matches phonetically by similarities in spelling.

Proprietary algorithms

Several organizations offer data scrubbing, de-duplication, data normalizing, and merge-merge algorithms which implement some form of approximate text matching, albeit with varying degrees of success. These systems usually used are often proprietary, and seldom documented. The applicability of these systems must be carefully analyzed before adopting any one solution.

Other sources

The above list of alternative matching algorithms is far from complete, and provides only a beginning. This section lists other sources of matching, and approximate text matching, as well as links to additional Double Metaphone and other sources of phonetic matching.
Additional Double Metaphone implementations

- Links to several Metaphone and Double Metaphone implementations, including C, F

Additional Phonetic Matching/Approximate text matching resources

- "Phonetic String Matching: Lessons from Information Retrieval" by Zobel and Dart. comparing all of the matching techniques discussed in this article (except Metapho unfortunately), and a few more. Includes tables containing quantitative test results
- "A Guided Tour to Approximate String Matching" by Gonzalo Navarro. - An exceller string matching, which is different from, but related to, phonetic matching.
- "Approximate Text Searching" by Gonzalo Navarro. - A very exhaustive discussion matching issues. May be too technical for some readers. Note that the English vers text.
- "Searching Proper Names in Databases" by Pfeifer, Poersch, and Fuhr. - A very acc several techniques specifically designed for searching databases of proper (last) na efficacy tests by the authors.
- "Learning String Edit Distance " by Ristad, and Yianilos. - Paper describing edit dist presenting an interesting stochastic model for learning an edit distance function frc latter subject is likely to be of limited interest to those seeking approximate string however the former topic is highly relevant.
- Richard Birkby's CodeProject article presenting four Soundex variations in C#. - Inc Soundex performance.
- A very readable essay by Michael Gillel and describing the Levenshtein Distance alç code in VB, C++, and Java. Be sure to take note of the Resources section, which ir sites and implementations.
- "Finding String Distances", Dr Dobb's Journal, April 1992, by Ray Valdes. Interestir Levenshtein distance, and its applications not only to string matching but also to m handwriting recognition, etc. Article available on DDJ archive CD. Source code avail
- "A Comparison of String Distance Metrics for Name-Matching Tasks" by Cohen, Ra nice, technical survey of techniques for measuring the "distance" between strings, matching proper names. Great source of additional information on edit distance tec Levenshtein distance.
- SourceForge home for SecondString, a Java approximate string matching library b\ Comparison of String Distance Metrics for Name-Matching Tasks" referenced above things, an implementation of Levenshtein distance.
- A powerful full-text search engine implemented entirely in Java. - Includes Levens with all of the typical full text indexing features.
- AGREP - AGREP is a tool, similar to egrep, fgrep, and grep, which searches for a gi files. AGREP uses an edit distance algorithm to perform approximate matching, ma experimenting with the results one can expect from such algorithms.

Conclusion

This article concludes the article series on phonetic matching of name data with Double Me some of the other major techniques for phonetic matching are presented, as well as links for the reader interested in further research on the subject. By this point, the reader sho one solution exists to the problem of matching similar but not identical text, and that car selected based on the reader's specific criteria.

That said, hopefully this article series will lead the reader to strongly consider Double Me of use, respectable performance, and readily available implementations in a variety of lai

History


3/13/2007
About Adam Nelson

My name is Adam Nelson. I've been a professional programmer since 1996, working in development, early first-generation web applications, modern n-tier distributed applications, security tools, to my last job as a Senior Consultant at BearingPoint posted in Bagdad. I currently work as a Senior Consultant at AppA, a Virginia startup developing super-secret tools and generally having a lot of fun.

I have a wide range of skills and interests, including cryptography, image processing, military history, 3D graphics, database optimization, and mathematics, to name a work (either for my employer or on my own self-education projects), read, and try to make sense of the world. Click here to view Adam Nelson's online profile.

Other popular String articles:

- **The Complete Guide to C++ Strings, Part II - String Wrapper Classes**
  A guide to the string wrapper classes provided by Visual C++ and class libraries.

- **The Complete Guide to C++ Strings, Part I - Win32 Character Encodings**
  A guide to the multitude of string types used in Windows.

- **CString-clone Using Standard C++**
  A Drop-In replacement for CString that builds on the Standard C++ Library's basic_string template.

- **CString Management**
  Learn how to effectively use CStrings.

[Top]

Sign in to vote for this article:  

Poor

Note: You must Sign in to post to this message board.

FAQ

Message score threshold 3.0

View Message View  

Per page 25

Msgs 1 to 23 of 23 (Total: 23) (Refresh)

Subject

- **SOUNDEX**
- **Re: SOUNDEX**
- **I can't register the MetaphoneCOM.dll**
- **Re: I can't register the MetaphoneCOM.dll**

Author

- **RichardGrimm**
- **Adam Nelson**
- **chuijhk**
- **Adam Nelson**


3/13/2007
Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

Re: Has Anybody got this to work?

I have a working Engine

Another reference to add to your list

Re: Another reference to add to your list

Implementation

Re: Implementation

Re: Implementation

Re: Implementation

Last Visit: 16:14 Tuesday 13th March, 2007