Fast and Effective Searches of Personal Names in an International Environment

Dr.Sandeep Gupta
Department. of CSE,
Noida Institute of Engg. Technology,
Greator Noida, India

Arun Pratap Srivastava
Department. of CSE,
G.L.Bajaj Institute of Technology &
Management, Greator Noida, India
arun019@yahoo.com

Dr. Shashank Awasthi
Department. of CSE,
G.L.Bajaj Institute of Technology
& Management, Greator Noida,
India

ABSTRACT

Fast and effective search of personal names in an international environment uses concepts of approximate string matching and applies them to special case of finding ‘close’ or ‘similar’ names, to an input name, from a large database of names. Such Proper-Name-Approximate matching finds applications in situations where a user is unsure of how a person’s name is spelled, such as in a telephone directory search system or a library search system where a user wishes to search books on an author’s name.

In this Paper we examine this problem in two main aspects: How to organize data efficiently, so as to obtain relevant results quickly, and how to develop suitable search techniques which would rank results suitably. We suggest four new data organization techniques to replace the current standard technique, Soundex, and we suggest refinements to the currently available search techniques. We then assess the performance of the developed techniques and compare them against the currently available ones.

Keywords

Hash table, Editex, Q-grams, Soundex, Hindex

1. INTRODUCTION

Finding the occurrence of given input string from a very large dataset is a fundamental problem in computer science. Simple string matching is the process of identifying a string or substring in a dataset (such as text) which is same as the input. It finds applications in various fields such as text processing & bioinformatics. Approximate string matching, however, involves finding strings (and/or substrings) which may not be exactly same as the input, but be ‘similar’ to the input string. A very frequently used application of approximate string matching is an automatic spelling suggestion program where a user is presented with ‘closely similar’ words to the erroneous word. Other such applications include studying gene mutations, identifying subsequences in data, virus & intrusion detection, file comparison, optical character recognition, etc. There are two important issues, which are to be considered while developing such system: Speed of result retrieval and Precision of results. The question regarding speed is largely one of data organization. If the dataset is suitably organized it would be easier to eliminate totally irrelevant results and retrieve only those results which are ‘good’ matches. Of course data organization also influences the ‘recall’ of the result set. This means that depending upon the selection scheme used to eliminate irrelevant results, some of the ‘relevant’ results might be lost. Since ‘recall’ is defined as the ratio of (a) number of relevant results retrieved to (b) the total number of relevant results; results missed out due to the selection scheme adversely affect ‘recall’. We present here six different data organization schemes which were explored. After such a limited set is considered we need to identify among this set, which are the results, the user would ‘approve’ as a ‘similar sounding match’. The ’precision’ of the system could then be defined as the ratio between (a) common number of matches obtained between to sets: the one which the user deems as ‘approved’ and the one produced by the system and (b) the total number of results retrieved. Thus, we need a system which has a high ‘precision’ value. We present here seven such search techniques which would provide fairly precise results. Following are the different methods by which the database could be formed:

- No Rep and order
- No rep no order
- Rep with order
- Rep no order
- Soundex
- Q-grams

The first four methods are somewhat similar-since the all have the same basic concept of removing the vowels in a given name unless the name starts with a vowel. The point where they differ is, while forming groups in some methods either repetitions or the order in which the consonants occur or both are considered. Once the database is formed one of the following search methods is used to retrieve the set of possible answers

- Edit Distance
- Edit Distance Tapered
- Editex
- Editex Tapered
- Ipadist
- Ipadist Tapered
- Q-gram

1.1 Data structure

- Hash table organization is used to store the names.
- Hash table has the following organization

Key1 => Value 1

Key2 => Value 2a,Value 2b, …

- So, for each name in the text file some processing is done and for a key generated the corresponding name is stored as one of the value.

1.2 Design

The following diagram represents the flow of control adopted in the retrieval of the answers required.

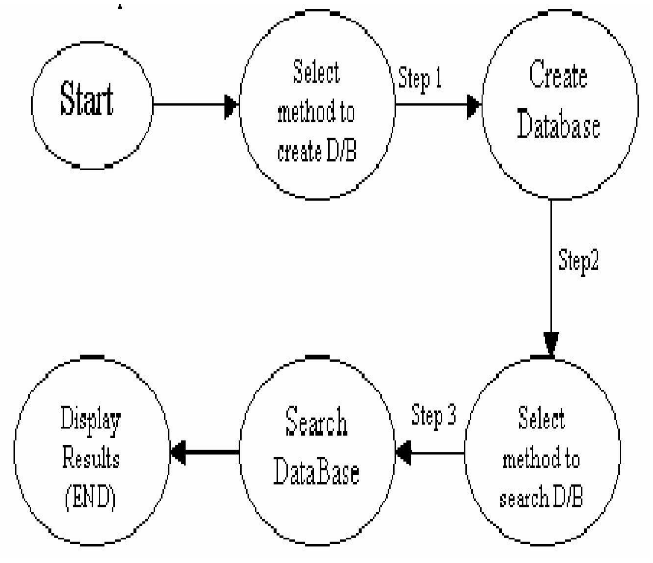

Figure 1: Flow of control

2. DATABASE ORGANIZATION

2.1 Database Creation

2.1.1 No repetition and order important

Step 1: All the vowels are removed except if at occurs at position 1 in the word.

Step 2: All the double occurrences in the word so formed are removed.

This method takes care of spelling mistakes.

2.1.2 No repetition and order not important

Step 1: All the vowels are removed except if at occurs at position 1 in the word.

Step 2: All the double occurrences in the word so formed are removed.

Step 3: The ordering of the intermediate key, formed from step 2 is removed by arranging the letters in ASCII order.

This method takes care of spelling mistakes and increases the scope of search

2.1.3 Repetition allowed and order important

Step 1: All the vowels are removed except if at occurs at position 1 in the word. This is the only step required to obtain the key.

This method produces specific results.

2.1.4 Repetition allowed but order not important

Step 1: All the vowels are removed except if at occurs at position 1 in the word.

Step 2: The ordering of the intermediate key so formed from step 1 is removed by arranging the letters in ASCII order

This method is slightly more scope of search than the previous method since the ordering is not considered.

2.1.5 Soundex

Soundex is a phonetic algorithm for indexing names by their sound. The basic aim is that the names with the same pronunciation to be processed to a same string so that matching can occur despite minor differences in spelling.

- The method relies on generating a code for each word.
- The Soundex code for a name consists of a letter followed by numbers: the letter is the first letter of the name, and the numbers contain information about the remaining consonants. Similar sounding consonants share the same number

The exact algorithm is as follows:

1. Retain the first letter of the string
2. Remove all occurrences of the following letters, unless it is the first letter: a, e, i, o, y, w, h, y.
3. Assign numbers to the remaining letters (after the first) as follows:

Soundex Code:	0	1	2	3	4	5	6
Letters:	aeiouhw	bpfv	cgjkqsxz	dt	l	mn	r

Figure 2 : Soundex Codes

4. If two or more letters with the same number were adjacent in the original name (before step 1), or adjacent except for any intervening h and w then omit all but the first.

2.1.6 Q-Grams

- This method aims at creating multiple keys each of length "3" from a given single name and then placing the name in each of the keys created.
- The method's underlying principle is that all the letters are important.
- Here, for given name, three consecutive letters are taken at a time (also called **tri-gram**) – starting from the first letter till the last letter is a part of a tri-gram.

3. SEARCHING TECHNIQUES

3.1 Searching Methods

The next step in system design is implementation of searching methods. Once the database is created with any of the six data organization methods detailed above, one of the following search methods is used to search the user input against the database created

3.1.1. Edit Distance
3.1.2. Editex
3.1.3. Q-Grams
3.1.4. Edit Distance Tapered
3.1.5. Editex Tapered
3.1.6. Ipadist
3.1.7. IpaDist Tapered
3.1.8. Hindex

3.1.1 Edit Distance

Edit Distance also known as Levenshtein distance (LD) is a measure of the similarity between two strings, which we will refer to as the source string (s) and the target string (t).The distance is the number of deletions, insertions, or substitutions required to transform source string into target string.

For example

- If s is "test" and t is "tent", then LD(s,t) = 1, because one substitution (change "s" to "n") is sufficient to transform s into t.
- If s is "test" and t is "test", then LD(s,t) = 0, because no transformations are needed. The strings are already identical.

The greater the Levenshtein distance, the more different the strings are.

The Algorithm

edit (0,0)=0, edit (i,0)=i, edit (0,j)=j
edit (i,j)=m in[edit(i-1,j) +1, edit(i,j-1) + 1,
edit(i-1,j-1) + r (s i ,t j)]

Figure 3: Recurrence relation for minimal edit distance

3.1.2 Editex

Editex is a phonetic distance measure that combines the properties of edit distances with the letter-grouping strategy used by Soundex and Phonix.
Editex is defined by the edit distance recurrence relation of Fig 4 with a redefined function r(a, b) and an additional function d(a, b).

edit (0,0)=0
edit (i,0)=edit(i-1,0)+d(s i-1 ,s i)
edit (0,j)=edit(0,j-1)+d(t j-1 ,t j)
edit(i,j)=min[edit(i-1,j)+d(s i-1,s i),edit(i,j-1)+d(t j-1 ,t j), edit(i-1,j-1) + r (s i ,t j)]

Figure 4: Recurrence relation for minimal editex

The function r(a, b) returns 0 if a and b are identical, 1 if a and b are both occur n the same group, and 2 otherwise. The groups are listed below:

Editex Code:	0	1	2	3	4	5	6	7	8	9
Letters:	aeiouy	bp	ckq	dt	lr	mn	gj	fpv	sxz	csz

Figure 5: Editex code groupings

The function d(a, b) is identical to r(a, b)—thus allowing pairs of the same letter to correspond to single occurrences of that letter--except that if a is h or w (letters that are often silent) and a = b then d(a, b) is 1.

3.1.3 Q-Grams

Q-Gram method for searching is almost similar to the edit distance method except that the comparisons in q-grams are made in groups rather than one letter at a time as in the case of edit distance.

- Q-grams are string distance measures based on q gram counts, where a q-gram of string s is any sub string of s of some fixed length q.
- A simple such measure is to choose q and count the number of q-grams two strings have in common.
- However, simply counting q-grams does not allow for length differences; for example, Fred has exactly as many q- grams in common with itself as it does with Frederick. So to address the problem, an q-gram distance which for strings without repeated q-grams (q-gram repeats are rare in names) can be

defined as

|Gs| + |Gt| - 2|Gs□Gt|

Where,

Gs: Set of q-grams in string s

Gt: Set of q-grams in string t

Gs□Gt: Set of q-grams common to Gs & Gt

Although this formula gives us a q-gram distance it does not tell us a percent match between two strings.

3.1.4 Edit Distance Tapered

Tapering is a refinement to the edit distance technique.

- It is based on the human factors property: “Differences at the start of a pronunciation can be more significant than differences at the end”.
- A tapered edit distance of particular interest is one in which the maximum penalty for replacement or deletion at the start of the string just exceeds the minimum penalty for replacement or deletion at the end of the string.
- Such an edit distance, in effect, breaks two ties : two errors always attract a higher penalty than one, regardless of position, but strings with one error are ranked according to the position at which the error occurs.

3.1.5 Editex Tapered

- The same tapering scheme is applied to the Editex method.
- The values obtained are nearly three times those obtained edit distance tapered algorithm.

3.1.6 IpaDist

- IpaDist is a phonometric search method developed by Justin Zobel (RMIT, Australia) and Philip Dart (University of Melbourne, Australia)
- IPA is the International Phonetic Algorithm. The strings are converted into phonetic codes as defined by the IPA.
- The codes, called phonemes are then compared by assigning distance values between different phoneme pairs. An editex like algorithm is used.

3.1.7 IpaDist Tapered

- The tapering scheme when applied to IpaDist gives us this modified method.
- This method, however, seems to give us inaccurate results.

3.1.8 Hindex

- Transliteration refers to the conversion of a string from one language to another. (E.g. English to Hindi)
- It is important to capture the pronunciations in the native language of the name
- To find the Hindex distance between s1 and s2 we first convert the consonant/consonant groups of both the strings into their Unicode Hindi representation based on Harvard-Kyoto transliteration scheme, a standard ‘English to Hindi Transliteration scheme’.

- The character groupings are as follows:

क ख ग घ ङ
k kh g gh G

च छ ज झ ञ
c ch j jh J

ट ठ ड ढ ण
t Th D Dh N

त थ द ध न
t th d dh n

प फ ब भ म
p ph b bh m

य र ल व
y r l v

श ष स ह
z S s h

- We apply the Editex algorithm by but use the above character groupings rather than the standard ones.
- We replace the all instances of the ‘d’ function by the ‘r’ function.
- His modification of Editex is termed:Hindex

3.2 Gram analysis

- The method takes into account the various points at which possible errors occur during pronunciation to representation.
- Variants of the same name can be identified by suitable analysis to find 2-grams or 3-grams which could be possibly misspelled or confused for the same pronunciation e. g

  aa a

  ph f

  sh s

  th t

  ky ki ………..

  ci si

  ava av

  aks ax

  etc,

- This means that the initially occurring n-gram can be replaced safely without altering the pronunciation, much.
- Any search method is used after the n-gram analysis function is applied when this particular search method altering technique is chosen.

4. METHODS FOR PERFORMANCE ASSESSMENT

A test bed of 28 queries was created. A spelling mistake was purposely introduced in these queries. For each such test query the entire database was manually scanned and the results deemed ‘relevant’ for this query were noted. Performance metrics for each of the 28 queries were obtained, and the average of these for a particular hash organization-search scheme combination was calculated. The metrics used were:

4.1 Recall

It is the ratio of the relevant results retrieved to the total number of relevant results (in a pre-defined result set).
It is a measure of ‘false negatives’, i.e. it is also an indicator of which results were marked as ‘irrelevant’ but were supposed to be marked as ‘relevant’ by the system.

$$\textbf{Recall} = \frac{|\{\text{relevant doc.}\}\cap|\{\text{retrieved doc.}\}}{|\{\text{relevant doc.}\}|}$$

4.2 Precision

- It is the ratio of the ‘relevant results’ (from the retrieved result set) to the total number. of results retrieved.
- It is a measure of ‘false positives’, i.e. it is also an indicator of which results were marked as ‘relevant’ but were supposed to be marked as ‘irrelevant’ by the system.

$$\textbf{Precision} = \frac{|\{\text{relevant doc.}\}\cap|\{\text{retrieved doc.}\}}{|\{\text{relevant doc.}\}|}$$

4.3 Weighted Recall

Since some results were deemed to be of more importance than others, a weighted recall scheme was considered:

- Each ‘relevant result’ set was divided into subsets. The no. of subsets for each query was decided individually based on the need to grade some results more extremely important than other subsets.
- For 4 subsets, the subset weights were, Set 1: 40%, Set 2: 30%, Set 3: 20%, Set 4:10%
- For 3 subsets, Set 1: 50%, Set 2: 30%, Set 3: 20%
- For 2 subsets, Set 1: 70%, Set 2: 30%

Such a weighted-scheme of assessment was considered more reliable as a performance metric than simple recall.

4.4 Time analysis

The time taken to execute a query is of utmost importance to measure the effectiveness of a data organization scheme. For each combination of ‘data-organization’ and ‘search scheme used’ the average time to execute a query was calculated by obtaining the run-times of 28 (different length) queries and then averaging the values obtained.

5. CONCLUSION

From the calculation of the weighted recall values, it is seen that the data organization techniques have a large role to play in the ‘quality’ of results obtained. The ‘no-rep-order’ data organization scheme provides the best recall values for all the search schemes. This implies that the order of the consonants in the names has an important effect on the pronunciation. Also if the same consonant occurs consecutively, the extra occurrence(s) can be safely overlooked. This scheme also has an optimal words/bin size, which is better than Soundex or Phonix, which are the generally used methods for grouping similar sounding names. Thus we suggest the ‘no-rep-order’ as a newer and better data organization scheme.

REFERENCES

[1] Finkel, Jenny Rose, Grenager, Trond and Manning, Christopher. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics ACL 2005), pp. 363-370.

[2] Malouf, Robert. 2002 Markov models for language independent named entity recognition. In Proceedings of CoNLL-2002 Taipei, Taiwan, pages 591-599.

[3] Justin Zobel and Philip Dart “Phonetic String matching: lessons from Information Retrieval”, SIGIR'96,Zurich ,pp. 105-110, 1996.

[4] Pattern Matching Algorithms, Alberto Apostolico & Zvi Galil, Oxford University Press, UK, 1997.

[5] R. Baeza-Yates and G. Navarro. Fast Approximate String Matching in a Dictionary. Proc. 1998.

[6] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10 (1966)

[7] Zobel, J. and Dart, P. [1995]. Finding approximate matches in large lexicons.Software-Practice and Experience, 25(3):331-345.

[8] Zobel, J. and Dart, P. [1996]. Fnetik: An integrated system for phonetic matching. Technical Report 96-6, Department of Computer Science, RMIT.