Edit distance computation with minimum number of edit operations in database management system and information retrieval

Yi Mar Myint*
*Faculty of Information Science, University of Computer Studies, Monywa

DOI: 10.29322/IJSRP.9.09.2019.p9385
http://dx.doi.org/10.29322/IJSRP.9.09.2019.p9385

Abstract- *In information theory, linguistics and computer science, the Levenshtein distance is a string metric for measuring the difference between two sequences. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other. The edit distance between two sequences is the minimum number of weighted transformation-operations that are required to transform one string into the other. The weighted transformation-operations are insert, remove, and substitute. Dynamic programming solution to find edit distance exists but it becomes computationally intensive when the lengths of strings become very large. And Edit distance is also a way of quantifying how dissimilar two-strings (eg. words) are to one another by counting the minimum number of operations required to transform one string into the other in computational linguistics and computer science. A database management system (DBMS) is a software package with computer programs that control the creation, maintenance, and use of a database. Information retrieval emerged as independent researchers are from traditional database management system more than a decade ago. This was driven by the increasing functional requirements that modern full text search engines have to meet. Current database management systems are not capable of supporting such flexibility. However, with the increasing of data to be indexed and retrieval and the increasing heavy workloads model such engines suffer from scability, reliability, distribution and performance problem. This paper implements the key to compute the minimum operations of edit distance or Levenshtein distance between the two strings. The benefits are provided to reduce the operations, full text search engines and spelling correction.*

Index Terms- Edit distance, Full text search engines, DBMS IRS, Spelling correction, inverted index;

I. Introduction

Nowadays, we have access to huge digital information. For this reason we need an efficient search process to obtain relevant information to a specific query. Taking into account that there is a greater number of documents to process and that we need greater precision and accuracy in the results obtained, the information retrieval process is everyday more costly and difficult. The relational data model is widely accepted as a high level interface to classical (formatted) data management. It turns out, however, to be inconvenient for handling even simple data structures as commonly used in information retrieval systems.

A database management system (DBMS) is a software package with computer programs that control the creation, maintenance, and use of a database. It allows organizations to conveniently develop databases for various applications by database administrators (DBAS) and other specialist [1].

The meaning of the term information retrieval can be very broad. Just getting a credit card out of your wallet so that you can type in the card number is a form of information retrieval. However, as an academic field of study, information retrieval might be defined thus: Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers). Information retrieval is fast becoming the dominant form of information access, overtaking traditional database-style searching. The term unstructured data refers to data that does not have clean, semantically overt, easy-for-a computer structure. It is the opposite of structured data. IR is also used to facilitate semi structured search as finding a document where the title contains java and the body contains threading. This is definitely true of all text data if you count the latent linguistic structure of human languages. But even accepting that the intended notion of structure is overt structure, most text has structure, such as headings and paragraphs and footnotes, which is commonly represented in documents by explicit markup (such as the coding underlying web 1. The field of information retrieval also covers supporting users in browsing or filtering document collections or further processing a set of retrieved documents [2].

Information retrieval techniques have been traditionally exploited outside of relational database systems, due to storage overhead, the complexity of programming them inside the database system, and their slow performance in SQL implementations. The key concept of this approach is to enhance and leverage enterprise IT technologies and provide a database-centric solution for image data management as well as key image processing operations. It not only offers the aforementioned standard database benefits but also goes one step further to provide more advantages to tackle the specific requirements derived from the two major challenges facing the geoimaging and geospatial industry. This project supports the idea that searching and querying with information retrieval models in relational database systems can be performed with optimized

SQL queries and User-Defined Functions. In our project, we proposed several techniques divided into two phases: storing and retrieving. The storing phase includes executing document pre-processing, stop-word removal and term extraction, and the retrieval phase is implemented with three fundamental IR models [3].

Edit distances find applications in natural language processing, where automatic spelling correction can determine candidate corrections for a misspelled word by selecting words from a dictionary that have a low distance to the word in question. In bioinformatics, it can be used to quantify the similarity of DNA sequences, which can be viewed as strings of the letters A, C, G and T. In information theory, linguistics and computer science, the Levenshtein distance is a string matric for measuring the difference between two sequences. The Levenshtein distance can also be computed between it, which is roughly proportional to the product of the two string lengths, makes this impractical. Thus, when used to aid in fuzzy string searching in applications such as record linkage, the compared strings are usually short to help improve speed of comparisons.For example, the Levenshtein distance between "kitten" 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: kitten → sitten (substitution of "s" for "k") sitten → sittin (substitution of "i" for "e") sittin → sitting (insertion of "g" at the end)[4].

The Levenshtein distance between two strings of the same length is strictly less than the Hamming distance is given by the pair "flaw" and "lawn". Here the Levenshtein distance equals 2 (delete "f" from the front; insert "n" at the end). The Hamming distance is 4.This paper presents to solve Edit distance problem of string matching. This work will be focusing on the problem of checking how similar two strings are, in other words, how closely two strings resemble.The algorithm is based on resolving dependencies in the dynamic programming solution of the problem and it is able to compute each row of Edit distance table in parallel. In this way, it becomes possible to compute the complete table in min (m, n) iterations for strings of size m and n whereas state-of-the-art parallel algorithm solves the problem in max (m, n) iterations [5].

II. RELATED WORK

In this paper, we attempt to bring IR back to database management systems. We propose using commercial DBMS as backend to existing full text search engines. Achieving this, today's search engines directly gain more robustness, scability, distribution and replication features provide by DBMS. We plan on mapping the whole interval index structure into database logical schema instead of just taking the file chunk as the smallest building block .The result of the query execution is a list of document IDs which satisfy the predicate described in the query. The results are usually sorted according to an internal scoring mechanism using fuzzy query processing techniques. The score is an indication of the relevance of the document which can be affected by many factors. This opens the door for ontological searches and other semantically richer similarity searches [6].

This work intends to capture of similarity between phrases. The algorithm is based on a dynamic programming approach integrating both the Edit distance between parse trees and single-term similarity. Our work stresses the use of the underlying grammatical structure, which serves as a guide in the computation of semantic similarity between words. The algorithm is based on a dynamic programming approach integrating both the edit distance between parse trees and single-term similarity. Our work stresses the use of the underlying grammatical structure, which serves as a guide in the computation of semantic similarity between words. This proposal allows us to obtain a more accurate notion of semantic proximity at sentence level, without increasing the complexity of the pattern-matching algorithm on which it is based1. This proposal allows us to obtain a more accurate notion of semantic proximity at sentence level, without increasing the complexity or the pattern-matching algorithm on which it is based [7].

Levenshtein Distance algorithm is an algorithm proposed to calculate the distance between two strings. This algorithm is widely used in NLP for spell checking and other tasks. Moreover, this algorithm is very useful in bioinformatics, as it used to calculate pattern matching between two DNA or protein sequences. However the high computation cost of this algorithm prohibits it from very useful in practice large-scale scenarios. In this work, we decrease the executions time for long two sequences with two main techniques. Latent Dirichlet allocation(LDA) is popular topic modeling technique for exploring hidden topics in text corpora. Increasing, topic modeling needs to scale to larger topic spaces and use richer forms of prior knowledge, such as word correlations or document labels.[8].

The result of Levenshtein Distance is based on the perception that if a metric holds the edit distance between all prefixes the first string and all the prefixes of the second, thus find the distance between the two full strings as the last value calculate The objects that are compared usually program code, algorithms, computer files, text versions, or complex data structures [9].

III THEORETICAL BACKGROUND

A. Edit distance

In computational linguistics and computer science, edit distance is a way of quantifying how dissimilar two strings (eg. Words) are to one another by counting the minimum number of operations required to transform one string into the other. Edit distance find applications in natural language processing, where automatic spelling corrections for a misspelled word by selecting, words from a dictionary that have a low distance to the word in question. Dynamic programming solutions exist to find edit distance but it becomes computationally intensive when the lengths of strings become very large. Hence, a parallel algorithm can always help in finding the solution in reasonable time.

For example, Levenshtein Distance between kitten 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. Kitten'n sitten (substitution of s for k)
2. Sitten' n sittin (substitution of I for e)
3. Sittin' n sitting(insertion of g at the end)

Upper and lower bounds [edit]

The Levenshtein Distance has several simple upper and lower bounds. These include:
-It is at least the difference of the sizes of the two strings.
-It is at most the length of the longer string. It is zero if and only if the strings are equal.
-If the strings are the same size, the Hamming distance is an upper bound on the Levenshtein Distance.
- The Levenshtein Distance between two strings is no greater than the sum of their Levenshtein Distance from a third string (triangle inequality)

B. Inverted Index

An index is highly efficient cross-reference lookup data structure. In most search engines, a variation of the well-known inverted index structure is used. An inverted index is an inside-out arrangement of documents such that terms take center stage. An index is highly efficient cross-reference look up data structure. In most search engine, a variation of the well-known inverted index structure is used. An inverted index is an inside-out arrangement of documents such that terms take center stage. Each term refers to a set of documents. Usually, a B+ tree is used to speed up traversing the index structure. The index process begins with collecting the available set of documents by the data gatherer. The parser converts them to a stream of plain text. For each document format, a parser has to be implemented. In the analysis phase, the stream of data is tokenized according to predefined delimiters and a number of operations are performed on the tokens.

To allow ranked retrieval: in many cases you want the best answer to an information need among many documents that contain certain words.

To gain the speed benefits of indexing at retrieval time, we have to build the index in advance. The major steps in this are:
1. Collect the documents to be indexed.
2. Tokenize the text, turning each document into a list of tokens.
3. To make index construction more efficient, we represent terms as termIDs (instead of strings as we did in Figure 1.4), where each termID is a unique TERMID serial number. We can build the mapping from terms to termIDs on the fly while we are processing the collection; or, in a two-pass approach, we compile the vocabulary in the first pass and construct the inverted index in the second pass.

Some information retrieval researchers prefer the term inverted file, but expressions like index construction and index compression are much more common than inverted file construction and inverted file compression.
4. In a (non-positional) inverted index, a postings is just a document ID, but it is inherently associated with a term, via the postings list it is placed on; sometimes we will also talk of a (term, doc ID)pair as a posting.

C. Full text search engines

Full text search engines do not care about the source of the data or its format as long as it is converted to plain text. Text is logically grouped into a set of documents. The user application constructs the user query which is submitted to the search engine. The result of the query execution is a list of document IDs which satisfy the predicate described in the query. The results are usually sorted according to an internal scoring mechanism using fuzzy query processing techniques. The score is an indication of the relevance of the document which can be affected by many factors. The phonetic difference between the search term and the hit is one of the most important factors. Some fields are boosted so that hits within these fields are more relevant to the search result as hits in other fields. Also, the distance between query terms found in a document can play a role in determining its relevance. E.g., searching for John Smith, a document containing John Smith has a higher score than a document containing John at its beginning and Smith at its end. Furthermore, search terms can be easily augmented by searches with synonyms. E.g., searching for car retrieves documents with the term vehicle or automobile as well.

D. Spelling correction

Isolated-term correction would fail to correct typographical errors such as flew form Heathrow, where all three query terms are correctly spelled. When a phrase such as this retrieves few documents, a search engine may like to offer the corrected query flew from Heathrow. The simplest way to do this is to enumerate corrections of each of the three query term is correctly spelled, then try substitutions of each correction in the phrase.

E. Typical Operations

Complete index creation: This operation occurs usually once. The whole set of documents is parsed and analyzed in order to create the index from scratch. This operation can take several hours to complete.

Full text search: This operation includes processing the query and returning page hits as a list of document IDs sorted according to their relevance.

Index update: This operation is also called incremental indexing. It is not supported by all search engines. Typically, a worker thread of the application monitors the actual inventory of documents. In case of document insertion, update, or deletion, the index is changed on the spot and its content is immediately made searchable. Lucene supports this operation.

IV APPLICATION DESIGN

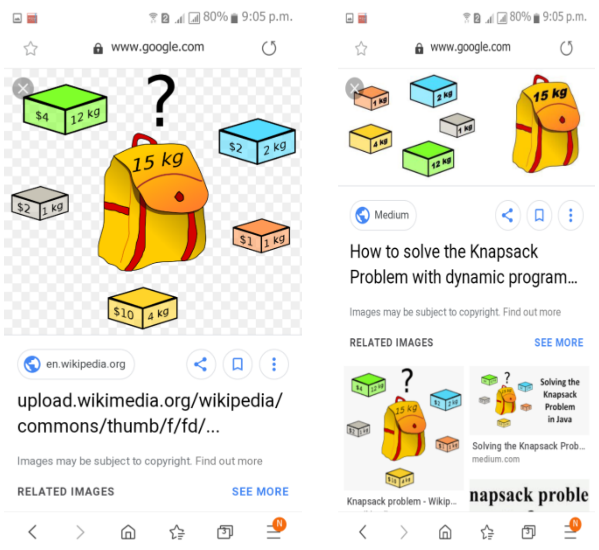

www.google.com
en.wikipedia.org
upload.wikimedia.org/wikipedia/commons/thumb/f/fd/...
Images may be subject to copyright. Find out more
RELATED IMAGES
SEE MORE
Medium
How to solve the Knapsack Problem with dynamic program...
Solving the Knapsack Problem in Java
Knapsack problem - Wikip...
napsack proble

Figure 1. Knapsack Problem with dynamic programming

In mathematics and computer science, **graph edit distance** (**GED**) is a measure of similarity (or dissimilarity) between two graphs. The concept of graph edit distance was first formalized mathematically by Alberto Sanfeliu and King-Sun Fu in 1983.A major application of graph edit distance is in inexact graph matching, such as error-tolerant pattern recognition in machine learning.

.The knapsack problem or rucksack problem is a problem in combinatorial optimization: Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.

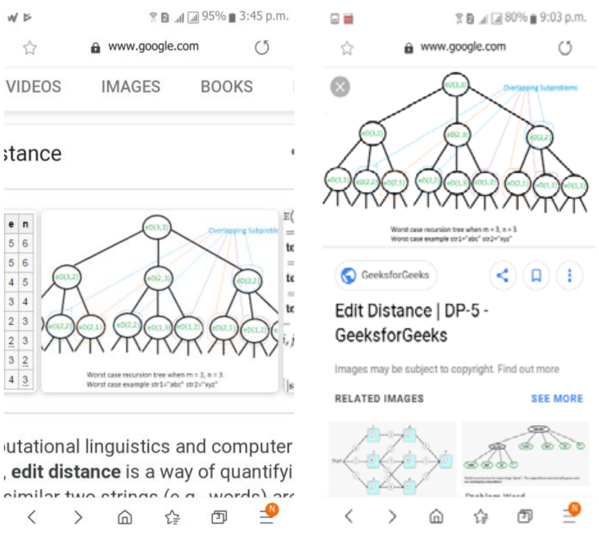
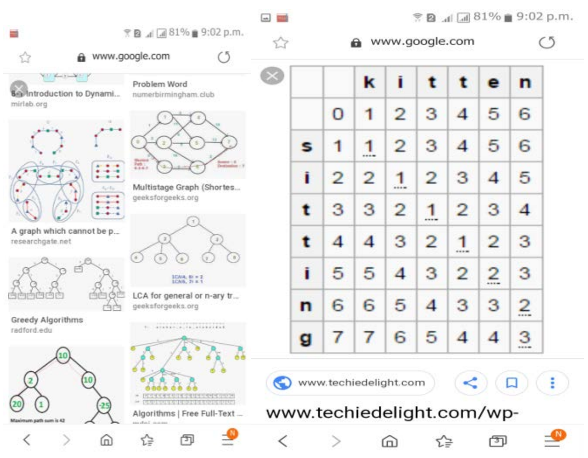

www.google.com
VIDEOS
IMAGES
BOOKS
stance
utational linguistics and computer
, edit distance is a way of quantifyi
GeeksforGeeks
Edit Distance | DP-5 - GeeksforGeeks
Images may be subject to copyright. Find out more
RELATED IMAGES
SEE MORE

Figure 2. Computational linguistics and computer, edit distance

In web search, the system has to provide search over billions of documents stored on millions of computers. Distinctive issues are needed to gather documents for indexing, being able to build systems that work efficiently at this enormous scale, and handling particular aspects of the web, such as the exploitation of hypertext and not being fooled by site providers manipulating page content in an attempt to boost their search engine rankings, given the commercial importance of the web.

Some information retrieval researchers prefer the term inverted file, but expressions like index construction and index compression are much more common than inverted file construction and inverted file compression.


www.google.com
Problem Word
Multistage Graph (Shortes...
A graph which cannot be p...
LCA for general or n-ary tr...
Greedy Algorithms
Algorithms | Free Full-Text ...
www.techiedelight.com
www.techiedelight.com/wp-

		k	i	t	t	e	n
	0	1	2	3	4	5	6
s	1	1	2	3	4	5	6
i	2	2	1	2	3	4	5
t	3	3	2	1	2	3	4
t	4	4	3	2	1	2	3
i	5	5	4	3	2	2	3
n	6	6	5	4	3	3	2
g	7	7	6	5	4	4	3

Figure 3. Levenshtein distance calculation researchgate.net

Levenshtein distance and theoretically compares our approach with state-of-the-art, and Section 5 discusses the implementation and experimental results.

Levenshtein distance or **edit distance** is the number of single character operations required to transform one string to another.

V IMPLEMENTATION OF EDIT DISTANCE

The Levenshtein's distance between two documents is defined as the minimum number of edit operations required to transform one text document into the other. The following edit operations are used by Levenshtein's edit distance algorithm to modify one document to another:

-Insertion
-Deletion
-Substitution

Example: Levenshtein's edit Distance between different strings:

right → fight (substitution of 'f' for 'r')

book → books (insert operation is performed at the end 's')

The Levenshtein's edit Distance between given strings depends on three basic operations to replace one string to another.

The results of Levenshtein’s distance is based on the perception that if a matrix holds the edit distance between all prefixes of the first string and all the prefixes of the second, Thus find the distance between the two full strings as the last value calculated.

Computing Techniques

- Dis(i,j) = score of best alignment from d11..d1i to d21…..d2j
- Dis (i–1, j–1) + d (d1i, d2j) //copy Dis (i–1, j) + 1 //insert
- Dis (i, j–1) + 1 //delete

Cost depends upon following factors:
Dis (0, 0) = 0 cost // if both strings are same
Dis (i, 0) = dis (i–1, 0) +1 = 0 // if source string is empty
Dis (0, j) = dis (0, j–1) +1 = 0 // if target string is empty

The algorithm:

Step 1: Initialization

I. Set a is the length of document 1 say d1, set b is length of document 2 say d2.

II. Create a matrix that consists 0 – b rows and 0 – a columns.
III. Initialize the first row from 0 to a.

IV. Initialize the first column from 0 to

Step2: Processing

I. Observe the value of d2 (i from 1 to a).

II. Observe the value of d1 (j from 1 to b).

III. f the value at d2[i] is equals to value at d1[j], the cost becomes 0.

IV. If the value at d2 [i] does not equal d1[j], the cost becomes 1.

V. Set block of matrix M[d1, d2] of the matrix equal to
1. The block immediately above add 1: M [d2–1, d1]
2. The block immediately to the left add 1: M [d2, d1–1] +1.
3. The block is diagonally above and to the left adds the cost: M [d2–1, d1–1] + cost.
Step 2 is repeated till the distance M [a, b] value is found.
Step 3: Result.

Computing Techniques

-Dis (i,j) = score of best alignment from d11..d1i to d21…..d2j
-Dis (i–1, j–1) + d (d1i, d2j) //copy Dis (i–1, j) + 1 //insert
-Dis (i, j–1) + 1 //delete

Cost depends upon following factors:
-Dis (0, 0) = 0 cost // if both strings are same
-Dis (i, 0) = dis (i–1, 0) +1 = 0 // if source string is empty

-Dis (0, j) = dis (0, j–1) +1 = 0 // if target string is empty.

Distance algorithm is a string edit distance algorithm which utilizes dynamic programming for its operation. Levenshtein distance is the minimum distance required to change one string into another. The change operations are insertion, substitution and deletion, applied on each string's elements sequentially. Fig. 1 showed two examples of calculating distance between strings using Levenshtein distance. The edit distance Ds,s' between two strings s and s' is described by the following recursive.

	Null	B	I	N
Null	0	1	2	3
P	1	1	2	3
I	2	2	1	2
N	3	3	2	**1**

Figure 4. The application of Levenshtein distance equation

Shows the application of Levenshtein distance equation, the edit distance between BIN and PIN is calculated as 1 i.e. if single character is replaced one word can be converted to another word. By replacing by B word PIN can be converted to BIN. Words are considered as similar if the edit or Levenshtein distance is less than threshold value. The threshold value can be determined by using the word length. Since Tri index is used for finding the edit distance if prefix (p) of a word exceeds the threshold value for edit distance then all words which contains p as prefix will be skipped from checking the edit distance which in turn help in determining similar words faster.

Levenshtein’s edit distance algorithm can be modified by removing the stop words. The words such as also, is, am, are, they, them, their, was, were etc. are ignored by search engine during processing are called stop word.

Most search engines are programmed to eliminate such words while indexing or retrieving as the outcome of search query. Stop words are considered inappropriate for searching purposes because they occur commonly in the language for which the indexing engine has been tuned. These words are dropped in order to save both time and space at the time of searching in the text documents. The words which are often used are is, am, are, they, them, also, the, of, and, to, in which are insignificant in IR and text mining. Stop words are removed to reduce due to following reasons:
-Each documents approximately consists 20–25% stop words.
-Efficiency of document is improved by removing the stop words.
-Stop words are not useful for searching or text mining
to reduce indexing.

	N	S	a	t	u	r	d	a	y
N	0	1	2	3	4	5	6	7	8
s	1	0	1	2	3	4	5	6	7
u	2	1	2	2	2	3	4	5	6
n	3	2	2	2	3	3	4	5	6
d	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	**3**

3 operations in total

Figure 5. The application of Edit distance equation

← OPERATION#1. DELETE a
← OPERATION#2. DELETE t
↖ OPERATION#3. TRANSFORM r to n

VI. CONCLUSION

In information Retrieval ranking the result set is very much important as the user will be interested in getting required information. Edit distance or Leveinshtein distance computation is a string metric for measuring the difference between two sequences. Levenshtein distance is used in fuzzy search. Proximity ranking makes use of modified inverted list by storing word positions in the form of bin Id. There are other advanced search features like auto completion, page ranking etc. which will be considered in future for implementation.

REFERENCES

[1] CJ.DATE,An Introduction to DATABASE SYSTEMS, 7th edition, canada, 2000
[2]http://www.cse.iitd.ac.in/~cs1150237/IIR/Sample_Output_IIR_02_Dec.txt
[3]https://www.researchgate.net/profile/Siva_Ravada/publication/252375540_AN_ENTERPRISE_DATABASECENTRIC_APPROACH_FOR_GEOSPATIAL_IMAGE_MANAGEMENT_AND_PROCESSING/links/0c96053ba7966389b2000000.pdf?origin=publication_list
[4]https://en.wikipedia.org/wiki/Edit_distance
[5]http://faculty.pucit.edu.pk/swjaffry/rpr/MURJET18EditDistance.pdf
[6]http://www.ijcsit.com/docs/Volume%203/Vol3Issue2/ijcsit2012030270.pdf
[7]http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.466.2522
[8]https://www.scholars.northwestern.edu/en/publications/efficient-methods-for-incorporating-knowledge-into-topic-models
[9]https://link.springer.com/chapter/10.1007/978-981-13-0755-3_6
[10] https://github.com/ctripcorp/C-OCR/tree/master/PostProcessing
[11]http://www.ubicc.org/files/pdf/IKE_BringingInformation_190.pdf
[12]https://howlingpixel.com/i-en/Graph_edit_distance
[13]https://www.ijser.org/researchpaper/Levenshtein-Distance-based-Information-Retrieval.pdf
[14] http://www.An Introduction to Information Retrieval, Christopher D. Manning PRabhakar Raghavan Hinrich Schiitze, Cambridge University Press Cambridge, England