International Journal of Psychosocial Rehabilitation, Vol. 24, Issue 06, 2020
ISSN: 1475-7192

API AND UI FOR TABLE LOOKUP APPROACH STEMMING ALGORITHM

[1]Jennifer .P, [2]Dr. A. Muthukumaravel

ABSTRACT-- *The system to retrieve the needed information or data from the google library or from the system library is completely depended on the technique which is called Information retrieval system. The proposed technique here, it retrieves the information on the supported text document currently present in the system library. If it is required, it can be set to any URL of the document. A unique glossary API built to set glossary, index and content. Uniquely here we used to google language library for to support multi-lingual. The UI is light weight and UI itself designed as an internal stub to create or retrieve data. In future it is easy to enable the voice input and entry points supports ANSI, ASCII, UNICODE and DBCS characters to support. The* **stopwords** *are the propositions or the vowels to avoid during the indexing of the words. The stemming has some unique methods to follow and to cut down the letters used.*

Keywords—*api,ui, table, Stemming algorithm*

I. INTRODUCTION

API stands for Application Programming Interface which is a set of protocols and definitions to be used by the technology for the proper communication between the application and the program used in the system via the internet. It actually allows the application to interact with the external service by using a simple set of commands. It is used to allow the developers to add some functionality to their applications to speed up the development process. API is used for faster development or to build the application and it acts like a building block. It becomes an integral part of application development. There is many main web APIs, but mainly used web service APIs. The web service APIs is a service oriented or software that uses the URL or www to provide access to its web services. The most commonly used web service APIs includes SOAP-Simple Object Access Protocol, XML-RPC, JSON-RPC-JavaScript Object Notation, REST-Representational State Transfer. We have used JSON in our proposed concept.

JSON-RPC It is a protocol that uses JavaScript Object Notation to transfer the data. JSON is a open standard file format derived from JavaScript. It is used by many other programming languages to include code to generate and parse data. JSON is essential and easier to work, it makes getting the most out of data a very simple process. Without JavaScript it is impossible to work with in this modern web development world. It develops like a small add-on to a multi-functional and strong working tool. JavaScript works and it helps the developers in all ways either from backend or frontend.

[1] *Research Scholar & Assistant Professor, Department of CS, Faculty of Arts & Sci., BIHER, Chennai, jennifer.mca@bharathuniv.ac.in.*

[2] *Dean-Faculty of Arts & Sci., BIHER, Chennai, muthukumaravel.mca@bharathuniv.ac.in.*

Received: 22 Sep 2019 | Revised: 13 Oct 2019 | Accepted: 15 Jan 2020

II. TABLE LOOKUP APPROACH ALGORITHM

Let f be the function to be implemented and I be the domain of interest. A typical table-lookup algorithm contains a set of "breakpoints" $_{Ck}$, k = 1,2, . . . , N in I and a table of N approximations T_k to f(c_k). For any input argument x E I, the algorithm calculates f(x) in three steps.

III. REDUCTION

For this given x, the algorithm selects an appropriate breakpoint c_k. It then applies a "reduction transformation".

r= R(x, C_k)

A typical case is R(x, ck) = x - C_k.

IV. APPROXIMATION

The algorithm now calculates f(r) using some approximation formula

p(r)≈ f(r)

V. RECONSTRUCTION

Based on the reduction transformation R, the values f(c_k) and f(r), and the nature of f, the algorithm calculates f (x) by a reconstruction formula S :

f(x) = S(f(c_k),f(r))

≈ S(f(c_k),p(r))

≈ S(T_k, p(r))

Although a traditional polynomial or rational-function based algorithm can also be expressed in these three steps, there are two properties peculiar to a table-lookup algorithm. First, the reduction process in a table-lookup algorithm is much more flexible since, unlike a traditional algorithm, the choice of breakpoints is basically independent of the function f in question. In most situations, the breakpoints are chosen so that the reduced argument r can be computed efficiently on the particular machine in question. Second, in a table lookup algorithm, the magnitude of the reduced argument r can be made as small as one wishes, limited only by the table size one can accommodate. We now illustrate these ideas by three realistic examples.

VI. ERROR ANALYSIS

Recall the three steps of calculating f at x:

Reduction: r = R(x, c_k).

Approximation: p(r) ≈ f(r)

Reconstruction: f(x) = S(f(c_k), f(r)) ≈ S(T_k, P(r))

Because of inexact computations, we obtain r instead of r, p instead of p, and S instead of S. Hence, the computed result is

Received: 22 Sep 2019 | Revised: 13 Oct 2019 | Accepted: 15 Jan 2020

$S(T_k, p(r))$

The goal of the error analysis is to estimate accurately the difference

$S(f(c_k),f(r)) - S(Tk, p(r))$

Time complexity of an algorithm quantifies the amount of time taken by an algorithm to run as a function of the length of the input. Similarly, Space complexity of an algorithm quantifies the amount of space or memory taken by an algorithm to run as a function of the length of the input. Time and space complexity depends on lots of things like hardware, operating system, processors, etc. However, we don't consider any of these factors while analyzing the algorithm. We will only consider the execution time of an algorithm.

Let's start with a simple example. Suppose you are given an array A, and an integer x and you have to find if x exists in array A.

Simple solution to this problem is traverse the whole array A and check if the any element is equal to x.

```
for i : 1 to length of A
   if A[i] is equal to x
      return TRUE
return FALSE
```

Time complexity notations

While analyzing an algorithm, we mostly consider O-notation because it will give us an upper limit of the execution time i.e. the execution time in the worst case.

To compute O-notation we will ignore the lower order terms, since the lower order terms are relatively insignificant for large input.

Let $f(N) = 2 * N^2 + 3 * N + 5$

$O(f(N))=O(2*N^2+3*N+5)=O(N^2)$

```
int count = 0;
    for (int i = 0; i < N; i++)
        for (int j = 0; j < i; j++)
     count++;
```

Let's see how many times **count++** will run.

When i = 0, it will run 0 times.

When i = 1, it will run 1 times.

When i = 2, it will run 2 times. and so on.

Total number of times **count++** will run is $(0 + 1 + 2 + \dots + (N-1) = \frac{N*(N-1)}{2})$. So the time complexity will be $O(N^2)$.

VII. EXECUTION

```
Public class LookUpTable{
Public static void main(String[] args){
String input = "Monday";
If(lookup(input)){
```

```
    JOptionPane.show MessageDialog(null, “Invalid”);
}else{
JOptionPane.show MessageDialog(null, “Invalid”);
} }
Public static boolean lookUp(String input){
String input = {“Monday”,”Tuesday”,”Wednesday”,”Thursday”,”Friday”,”Saturday”,”Sunday”};
boolean valid = false;
for(int i = 0; i < table.length; i++)
{ If(input.equals(table[i])) {
Valid = true;
} }
Return valid;
} }
```

If you RUN the above function it will show valid message, the string input value contains Monday this will cross check each other variable and shows valid or invalid message according to the input which is user gave. So accordingly keywords are stored it in a one variable finally we need to cross check with the original document which you uploaded. The final documents were generating by removing the stopwords.

```
Time  = microtime(true) = $_SERVER[“REQUEST_TIME_FLOAT”];
Return “Processtime:{Time}”;
```

Clustering and Extract the stopwords from the document

Processtime: 0.6154698

Time Complexity: 0.75468

Space complexity: 58%

Accuracy: 75%

VIII. CONCLUSION

The primary section manages the table lookup approach, which understand the linguistic standardization and decreased regular shape and remaining three classifications, for example, successor variety, affix removal, n-gram strategies additionally done likewise yet looking at and execution time may differ relies on the techniques and its properties. The second motivation behind a stemmer is specifically identified with the information retrieval process, as having the stems of the words permits a few periods of the information retrieval process to be enhanced, among which we can feature the capacity to list the reports as indicated by their points, as their terms are gathered by stems (that are like ideas) or the extension of a question to acquire an ever increasing number of exact outcomes.

REFERENCES

1. “Design and Development of a Stemmer for Punjabi” International Journal of Computer Applications, Dinesh Kumar, Prince Rana-Dec-10.

2. R.Shamili , J.Jeyaram, “Skirmish Against Password Denounce Using Graph Based Maze Generation Algorithm”, International Journal of Innovations in Scientific and Engineering Research (IJISER), Vol.4, no.4, pp.117-122, 2017.
3. Jennifer .P, Kannan Subramanian., “Retrieving the Personal Photos in Web Data” in International Journal of P2P Network Trends and Technology (IJPTT) – Volume2 Issue3 Number1 May 2012.
4. Composition of dynamic web service using petri-net, P. Jennifer, Dr.A.Muthukumaravel, 2015/2,
5. Mobile positioning technologies and location services, Jennifer.P, Dr.A.Muthukumravel, 2014
6. On-demand security architecture for cloud computing, K Sankar, S Kannan, P Jennifer, 2014 Middle-East J. Sci. Res
7. Prediction Of Code Fault Using Naïve Bayes And Svm Classifiers K Sankar, S Kannan, P Jennifer 2014
8. Ensuring Distributed Accountability for Data Sharing in Cloud K Karthick, P Jennifer, A Muthukumaravel 2014.
9. “A survey of Stemming Algorithms for Information Retrieval”, IOSR Journal of Computer Engineering (IOSR-JCE), Brajendra Singh Rajput, Dr. NilayKhare, June 2015
10. “Indexing Techniques on Information Retrieval”, International Journal of Psychosocial Rehabilitation, by Jennifer .P, A. Muthukumaravel, Vol. 24, Issue 01, 2020, ISSN: 1475-7192
11. “Indexing on IR System b y sing Stemming and Stopwords”, International Journal of Recent Technology and Engineering (IJRTE), by Jennifer .P, A. Muthukumaravel, ISSN: 2277-3878, Volume-8 Issue-1S2, May 2019
12. “Conflation Methods in Stemming Algorithm” , International Journal of Innovative Technology and Exploring Engineering (IJITEE), by Jennifer .P, A. Muthukumaravel, ISSN: 2278-3075, Volume-8, Issue-11S, September 2019
13. A Query Formulation Language for the Data Web” - IEEE Transactions on Knowledge And Data Engineering, Mustafa Jarrar and Marios D. Dikaiakos, Member, IEEE Computer Society-May-12.
14. “Multiagent Ontology Mapping Framework for the semntic web”-IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, Miklos Nagy and Maria Vargas-Vera-Jul-11.
15. “Toward SWSs Discovery: Mapping from WSDL to OWL-S Based on Ontology Search and Standardization Engine”-IEEE Transactions on Knowledge and Data Engineering, Tamer Ahmed Farrag, Ahmed Ibrahim Saleh, and Hesham Arafat Ali-May-13.
16. “The History of Information Retrieval Research”-Proceedings of the IEEE,Mark Sanderson and W. Bruce Croft-May-12.
17. “CONCEPT-BASED INDEXING IN TEXT INFORMATION RETRIEVAL” International Journal of Computer Science & Information Technology (IJCSIT), FatihaBoubekeur and Wassila Azzoug-Feb-13.