Automated Optimum Route Generator and Data Analyzer

Nitin Arora
Department of Informatics
School of Com. Sci
UPES, Dehradun

Somansh Garg
Department of Virtualization
School of Com. Sci
UPES, Dehradun

Varad Sant
Department of Virtualization
School of Com. Sci
UPES, Dehradun

Rohit Goyal
Department of Virtualization
School of Com. Sci
UPES, Dehradun

ABSTRACT

As a traveler, it has become a necessity to first select the most optimum analyze route that is the shortest path to travel the cities and then travel through the path. This paper is aimed to provide its users the most optimized path to travel, that will consume the minimum amount of time and distance. To achieve this, Travelling Salesman Problem algorithm through genetic algorithm has been implemented. The paper also provides with the feature of a data analyzer. This aims to analyze the data provided by the users for a particular city and generate the ratings based on the data. To provide the feature of data analyzer, Knuth-Morris-Pratt algorithm for pattern matching and TRIE data structure for efficient retrieval of data has been used. As every traveler always ensure the reviews and rating of the place before visiting it, getting a brief review and knowing the rating of the city will help them in selecting the city to visit.

Keywords

Route Optimization, Genetic Algorithms, Travelling Salesman Problem, Time Complexity, KMP Algorithm

1. INTRODUCTION

The paper aims to make the travel of users easy by providing them with the shortest path between several cities. The optimized path between the cities can be simply achieved by implementing algorithms. The paper is divided into two parts. The first part of the paper is aimed to calculate the most optimized path between the cities to be traversed. The paper implements the travelling salesman problem (TSP) by using genetic algorithm as it decreases time complexity significantly. The second part of the paper will deal with the data analysis i.e. the users can rate their visits in various cities and according to the reviews whether positive or negative the place will be rated that will be useful for other people planning out their trip in those cities. The Data analyzer part will be achieved by creating a database in the through files. To achieve this, various algorithms and various methods for searching of the word in the database by Knuth-Morris-Pratt's, Brute Force and other text searching algorithms are applied. The reviews entered by the user will be processed by separating its keywords and then accordingly it will be decided whether it's a positive or negative review. The processing of the raw data will be achieved by filtering and stemming algorithms. The main aim of the paper is to decide the optimum path for a set of given cities as selected by the user and help the user to decide those cities by generating the rating of the city based on the reviews given by the previous users of the application. Thereby the size of the database increases as the number of users increases making the application more reliable and vaster with the time. Its database will surely fulfill the customers' requirements and quench to know about the place better, as database will store enough number of reviews to help the user out and ensure all the breathtaking places present for the user.In today's era, people love travelling than anything else. Solo trips, family trips, biking are all different ways in which people travel for fun. They do not have a particular aim to visit cities they do it for purpose of exploring new places and sightseeing especially during the vacations. The software is aimed for these travelers only. Who will be telling the cities they want to visit and their starting location and which will be then used to generate the most optimum route between those cities for them. Apart from that they could search for a particular site or some unexplored places of the city in the application something which is not available on the internet. With help of reviews given for a particular city, traveler could decide which city to visit and which not to, also city ranking (automatically generated by the application) would further help the user in the selection of the city. These features will facilitate the users to plan their travelling.The main objective of the paper is to provide the most optimum route between a set of given cities where each should be travelled exactly once and should return to the starting city after travelling through all the cities and to provide search facilities over data of the cities, matching multiple words and substrings of the words and also to analyze the reviews of the cities to generate their rating.Rest of the paper is divided as follows: section 2 describes the literature review part, methodology part is discussed in section 3, Algorithms used and results are discussed in section 4, outputs and discussions in section 5 and are concluded in section 6 of the paper.

2. LITERATURE REVIEW

The paper basically aims to provide the shortest path to user between multiple cities. Several algorithms are used and implemented which are referenced from papers given below. Jean-Yves Potvin at al. [1] has proposed a survey of genetic algorithms for the traveling salesman problem. In this paper, a simple genetic algorithm is introduced, and various extensions are presented to solve the traveling salesman problem. Computational results are also reported for both random and classical problems taken from the operations research literature. Donald Knuth et al. [2] proposed a method known as Fast pattern in strings they developed an algorithm for pattern matching in string. The efficiency of the pattern matching is much faster than the algorithm that are proposed by others. James Inman [4] introduced a term haversine which is used to find the distance between two point using longitude and latitude.

This formula is used to find the distance between the cities. D*e la Briandais, Rene* [4-5] have introduced a new technique for file searching in their paper File sharing using variable length keys which led to introduction of the TRIE data structure by [6] *Black, Paul* in their paper Dictionary of algorithm and Data Structure.

3. METHODOLOGY

Genetic algorithms are stochastic search algorithms which act on a population of possible solutions. They are loosely based on the mechanics of population genetics and selection. The potential solutions are encoded as 'genes' – strings of characters from some alphabet. New solutions can be produced by 'mutating' members of the current population, and by 'mating' two solutions together to form a new solution. The better solutions are selected to breed and mutate and the worse ones are discarded. Genetic algorithms are used in artificial intelligence like other search algorithms which are used to search a space of potential solutions to find one which solves the problem. The paper is divided into two main parts that are Generation of the optimum path and Data analysis and search over the data.

3.1 Generation of the optimum path

The implementation consists of these five main parts described below –

Initialization of Chromosomes: Number of chromosomes denoted by 'n' and of length 'l' will be generated to initiate the algorithm and produce the first generation of the chromosomes where 'n' and 'l' will be pre-specified based on number of cities needed by the user.

Calculation of distance and normalization: A distance function will be used to calculate the distance between the cities i.e. chromosomes and these distances will be further normalized so that selection of chromosomes can be easily made for the next step.

Selection of Chromosomes form the pool: Chromosomes will be selected from the normalized pool based on their distance. Probability distribution will be used to increase the selection probability of the chromosomes with the least distance in order to get better offspring.

Genetic alteration: Genes of the chromosomes will be altered in the hope of producing even better offspring. Two main techniques - Mutation i.e. alteration of genes of the same chromosome and Crossover i.e. overlapping genes of one chromosome with other will be used

Creation of new generation: Newer generations will be produced by using the above- mentioned techniques in Genetic alteration. Number of generations to be produced will be predefined and depends on the number of cities selected by the user.

Data analysis and search over data

The data for analysis and searching will be based on the reviews provided by the user. The review will be treated as the raw input data by the system. The working on the raw data will be divided into following parts-

- Filtering of Raw data
- Processing of the filtered data

Filtering of Raw data – It is the process including elimination of stop words and process of stemming the words into their standard form for further processing. Manual filtering will be done by defining the data sets for stop words and root words for stemming.

Processing of the filtered data – A predefined dictionary of words will be used to identify the semantics of the words (data). This information generated will be further used to analysis of the raw data considered as a whole.Whereas searching will be done on the Raw input data to increase the vocabulary of the system and ensure that maximum number of matches is found for a given user query. Simple data search algorithm over text will be used.

4. ALGORITHM USED & RESULT DISCUSSION

4.1 Haversine Formula

Haversine formula is used to determine distance between two points in the sphere given their longitude and latitude. The formula (eq. 1) given below has been used to determine the distance between the cities with the help of their longitude and latitude [3].

$$hav(\theta) = hav(\varphi_2 - \varphi_1) + \cos(\varphi_1)\cos(\varphi_2)hav(\lambda_2 - \lambda_1)$$

(1)

Where, θ is the angle between two points, Φ1, Φ2 is the latitude of point 1 and point 2 respectively, $\lambda 1$, $\lambda 2$ is longitude of point1 and point 2 respectively. So, using the haversine formula the distance D (distance between two points) can be calculated by applying inverse sin or arcsine function to central angle [3].

$$d = r \times hav^{-1}(h) = 2r \times \arcsin(\sqrt{h})$$

(2)

Where, h is hav(Θ). On solving equation 1 and equation 2

$$D = 2r \times \arcsin(\sqrt{\sin^2(\frac{\varphi_2 - \varphi_1}{2}) + \cos(\varphi_1)\cos(\varphi_2)\sin^2(\frac{\lambda_2 - \lambda_1}{2})})$$

(3)

So, this is the haversine formula that has been used to find distance between two points on the sphere.

4.2 Knuth Morris Pratt (KMP) String Matching Algorithm

This is the algorithm used for pattern searching in a text. KMP algorithm is used because this algorithm because the matching time less than that of the other algorithm [7].

Table 1: Pre-processing time and matching time of different string matching algorithms

Algorithm	Pre-processing time	Matching Time
Naïve String Matching Algorithm	None	O(nm)
Rabin Karp Algorithm	O(m)	O(n+m)
KMP Algorithm	O(m)	O(n)

The main intention is to write a function to search pattern with two arguments pattern and txt and the basic idea of KMP is whenever a mismatch is detected, instead of scanning the word again it starts from the last matched suffix of the word, hence an efficient algorithm. So, to achieve this there need of so some preprocessing to store which of the word are matched.

Algorithm for pattern matching

```
Step 1: start
Step 2: creating an lps
//lps is array of size m i.e size of pattern which stores longest proper prefix which is also a suffix. int lps[M] //where m is length of pattern
Step 3: Computing lps
Unlike like other algorithm we use lps[] to compute the next pattern to match how can we achieve it
Step4: comparing pat[j] with j==0 with txt[i]
Step 5: keep matching txt[i] and pat[j] with increment in i and j till they keep matchin
Step 6: if mismatch found
Step 6a: we know that character pat[0…j-1] match with txt[i-j+1….i-1] so lps is lps[j-1] so we don't need to match this lps with txt[i-j…i-1]
Step 6b: Repeat step 6 till the txt file ends to find all the matches in the text file
Step 7: Fill the lps[] with the
pattern pat[0..m-1]
Step 8: Find the pattern in the string which is need to be found out.
Step9: end
```

4.3 Trie Data Structure & Operations

TRIE data structure is the data structure for efficient retrieval of data which stands for "Text Retrieval". Because of the use of the TRIE data structure the time complexity is reduced to O(M) from that of by using BST which is O(M*log N), M is maximum string length and N is no of keys in tree. Every node of TRIE has multiple branches and has a possible character of key. Every last node of every key as end of the word node this is done by isEndOfWord. Structure is required to represent tree

node ***struct*** *TrieNode* ***{***

Struct *TrieNode *children[ALPHABET_SIZE];*

*bool isEndOfWord; // to check if the node is end of a word****};***

4.3.1 Insertion

Every character of input key is as an individual TRIE node. Children is an array pointer to next level TRIENODE. The key character acts as an index to an array children. If input key is new or extension to existing key, a new node in tree is added if the key is prefix then there is no need to add the node just make the last node as end of word. The algorithm is as follows:

```
Step 1: start
Step 2: struct TrieNode *abc = root; //defining the root
Step 3: for(int i=0;i<key.length();i++) //start inserting
Step4: if(!abc ->children[index]) //check for the root
Step 5: abc ->children[index]=addnode()// new key or non existing key add new node
Step 6: abc -
>isEndOfWord
=true;// mark end of
the
Step 7: end
```

4.3.2 Searching

Searching is down from top to down approach. As this retrieve and tells if the word is present or not. There are two possibilities that the searching can stop

- can terminate due to lack or end of the string in this case if EndOfWord field of last node is true then key exist.
- if search terminate without examining all character of key, then key not present in trie.

```
Step 1: start
Step 2: struct Trienode *abc = root;
Step 3: for(int i=0;i<length();i++)//start searching
Step 4: if(!abc ->children[index])//searching fail 5.return false;
Step 6: else true;// if searching succeed
```

4.4 Travelling Salesman Problem using Genetic Algorithm

Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and returns to the starting point. Hamiltonian cycle problem is to find if there exist a tour that visits every city exactly once. Here, Hamiltonian Tour exists (because the graph is complete) and in fact many such tours exist, the problem is to find a minimum weight Hamiltonian Cycle. In this paper Genetic Algorithms is used solve the above stated problem. Genetic Algorithms (GAs) are adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic. The basic concept of GAs is designed to simulate processes in natural system necessary for evolution, specifically those that follow the principles first laid down by Charles Darwin of survival of the fittest. As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem. First pioneered by John Holland in the 60s, Genetic Algorithms has been widely studied, experimented and applied in many fields in engineering worlds. Not only does GAs provide an alternative method to solving problem, it consistently outperforms other traditional methods in most of the problems link. Many of the real-world problems involved finding optimal parameters, which might prove difficult for traditional methods but ideal for GAs. However, because of its outstanding performance in optimization, GAs have been wrongly regarded as a function optimizer. A simple and pure genetic algorithm can be defined in the following steps.

Step 1*. Create an initial population of P chromosomes.*

Step 2*. Evaluate the fitness of each chromosome.*

Step 3*. Choose P/2 parents from the current population via proportionalselection.*

Step 4*. Randomly select two parents to create offspring using crossover operator.*

Step 5*. Apply mutation operators for minor changes in the results.*

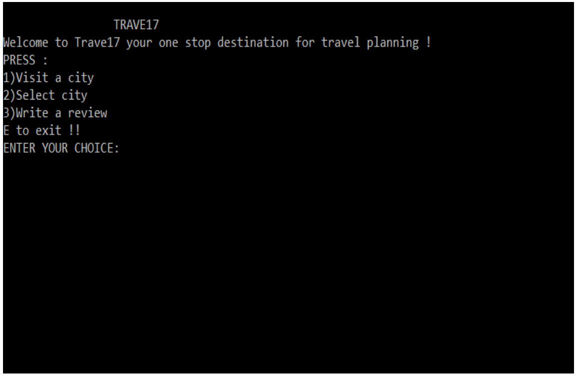
Step 6*. Repeat Steps 4 and 5 until all parents are selected and mated.*

Step 7*. Replace old population of chromosomes with new one.*

5. OUTPUTS AND DISCUSSION

Initially the traveler has three options as shown in the figure 1. He can select any one of them and they are:

- Select the cities the traveler wants to traverse.
- If the traveler needs to select city to travel, he shall go with this option as here the traveler will get the rating and reviews of all cities.
- The city traversed by the traveler can also be reviewed by the traveler.

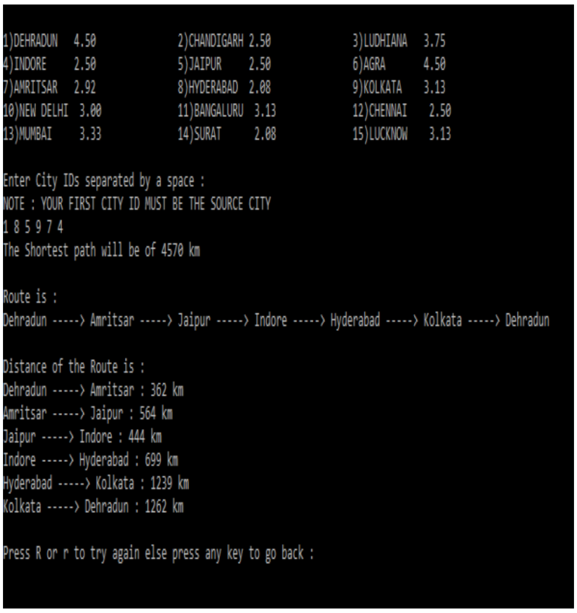


```
                 TRAVE17
Welcome to Trave17 your one stop destination for travel planning !
PRESS :
1)Visit a city
2)Select city
3)Write a review
E to exit !!
ENTER YOUR CHOICE:
```

Figure 1: Initial options with the traveler

5.1 Visit a City

The first option is to visit a city. In this, the user selects all the cities (figure 2), the paper also aims to provide user assistance for selecting the cities by providing them the rating of the cities alongside the name of cities. The paper results the shortest path between all the cities selected and then returning back to origin. Application also gives the exact distance between each pair of cities, the Haversine formula (eq. 1) is used, and it's the Genetic Algorithm that provides with the results of the shortest path between cities

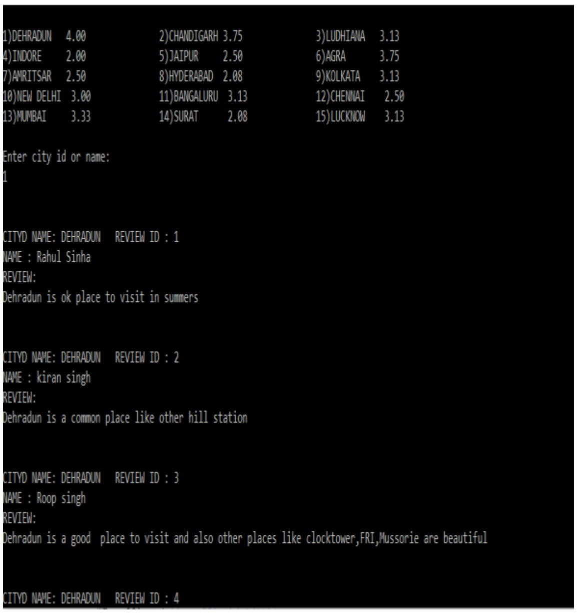


```
1)DEHRADUN   4.50          2)CHANDIGARH 2.50          3)LUDHIANA   3.75
4)INDORE     2.50          5)JAIPUR     2.50          6)AGRA       4.50
7)AMRITSAR   2.92          8)HYDERABAD  2.08          9)KOLKATA    3.13
10)NEW DELHI  3.00         11)BANGALURU  3.13         12)CHENNAI    2.50
13)MUMBAI     3.33         14)SURAT      2.08         15)LUCKNOW    3.13

Enter City IDs separated by a space :
NOTE : YOUR FIRST CITY ID MUST BE THE SOURCE CITY
1 8 5 9 7 4
The Shortest path will be of 4570 km

Route is :
Dehradun -----> Amritsar -----> Jaipur -----> Indore -----> Hyderabad -----> Kolkata -----> Dehradun

Distance of the Route is :
Dehradun -----> Amritsar : 362 km
Amritsar -----> Jaipur : 564 km
Jaipur -----> Indore : 444 km
Indore -----> Hyderabad : 699 km
Hyderabad -----> Kolkata : 1239 km
Kolkata -----> Dehradun : 1262 km

Press R or r to try again else press any key to go back :
```

Figure 2: City visit option

5.2 Select a city

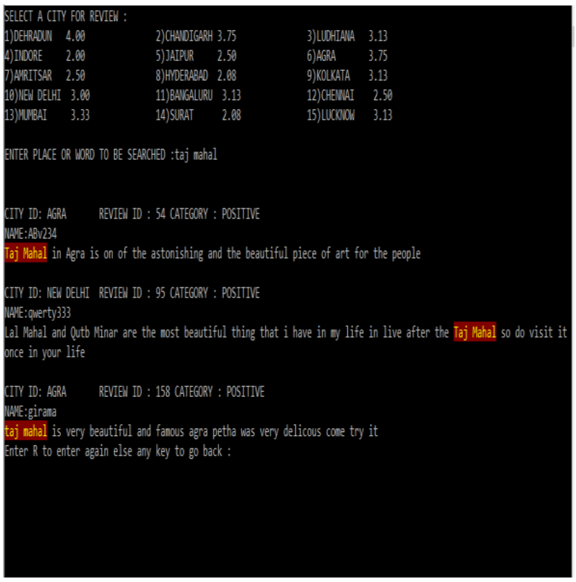
Another useful feature that the application provides to its user as the can see the rating and search the city by its id and see all the reviews entered by other travelers as shown in figure3.



```
1)DEHRADUN   4.00        2)CHANDIGARH 3.75        3)LUDHIANA   3.13
4)INDORE     2.00        5)JAIPUR     2.50        6)AGRA       3.75
7)AMRITSAR   2.50        8)HYDERABAD  2.08        9)KOLKATA    3.13
10)NEW DELHI  3.00       11)BANGALURU  3.13       12)CHENNAI    2.50
13)MUMBAI     3.33       14)SURAT      2.08       15)LUCKNOW    3.13

Enter city id or name:
1


CITYD NAME: DEHRADUN   REVIEW ID : 1
NAME : Rahul Sinha
REVIEW:
Dehradun is ok place to visit in summers


CITYD NAME: DEHRADUN   REVIEW ID : 2
NAME : kiran singh
REVIEW:
Dehradun is a common place like other hill station


CITYD NAME: DEHRADUN   REVIEW ID : 3
NAME : Roop singh
REVIEW:
Dehradun is a good  place to visit and also other places like clocktower,FRI,Mussorie are beautiful


CITYD NAME: DEHRADUN   REVIEW ID : 4
```

Figure 3: City selection with Review

This paper also provides the feature to select the city by keywords as shown in figure 4. The user gets the result as the keyword highlighted and all the reviews stored in the file structures in which the keyword has been used are displayed



```
SELECT A CITY FOR REVIEW :
1)DEHRADUN   4.00             2)CHANDIGARH 3.75             3)LUDHIANA   3.13
4)INDORE     2.00             5)JAIPUR     2.50             6)AGRA       3.75
7)AMRITSAR   2.50             8)HYDERABAD  2.08             9)KOLKATA    3.13
10)NEW DELHI  3.00            11)BANGALURU  3.13            12)CHENNAI    2.50
13)MUMBAI     3.33            14)SURAT      2.08            15)LUCKNOW    3.13

ENTER PLACE OR WORD TO BE SEARCHED :taj mahal


CITY ID: AGRA      REVIEW ID : 54 CATEGORY : POSITIVE
NAME:ABv234
Taj Mahal in Agra is on of the astonishing and the beautiful piece of art for the people

CITY ID: NEW DELHI  REVIEW ID : 95 CATEGORY : POSITIVE
NAME:qwerty333
Lal Mahal and Qutb Minar are the most beautiful thing that i have in my life in live after the Taj Mahal so do visit it
once in your life

CITY ID: AGRA      REVIEW ID : 158 CATEGORY : POSITIVE
NAME:girama
taj mahal is very beautiful and famous agra petha was very delicous come try it
Enter R to enter again else any key to go back :
```

Figure 4: City Selection using Keyword

5.3 Write a Review

The best feature of this application is Data Analysis and this includes the feature of review writing. The application provides a feature to traveler to write a review for cities already travelled by him. And the best part is the rating being dynamic that is the application itself recognizes whether a review entered is positive, negative or neutral and accordingly alters the rating. The Interesting part is the String Tokenizer recognizes the words in five categories that are positive, negative, amplifier, negation and rest all are considered as neutral that don't affect the rating. Once the tokenizer has identified the words it returns the Category of review entered whether the review is positive, negative or neutral along with this it returns the unique review id that is different for each review, and as the function defined earlier the user can check its entered review using its unique Review_Id.

6. CONCLUSION

This paper proposed process for ease the life of the traveler by providing the shortest path between the cities that he wants to travel. And will also help the traveler to make a choice that if he should or should not visit depending on the previous reviews from traveler.

7. REFERENCES

[1] https://iccl.inf.tu-dresden.de/w/images/b/b7/GA_for_TSP.pdf

[2] Knuth, Donald; Morris, James H.; Pratt, Vaughan (1977). "Fast pattern matching in strings". SIAM Journal on Computing. 6 (2): 323–350.doi:10.1137/0206024.

[3] Van Brummelen, Glen Robert (2013). Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry. Princeton University Press. ISBN 9780691148922. 0691148929. Retrieved 2015-11-10.

[4] Brass, Peter (2008). Advanced Data Structures. Cambridge University Press.

[5] Black, Paul E. (2009-11-16). "trie". Dictionary of Algorithms and Data Structures. National Institute of Standards and Technology. Archived from the original on 2010-05- 19.

[6] EH.L. Aarts, J.H.M. Korst and RJ.M. van Laarhoven, A quantitative analysis of the simulated annealing algorithm: A case study for the traveling salesman problem, J. Statist. Phys. 50(1988) 189 - 206.

[7] Garima Pandey, Mamta Martolia and Nitin Arora. A Novel String Matching Algorithm and Comparison with KMP Algorithm. *International Journal of Computer Applications* 179(3):6-8, December 2017.