MANAGING ROAD ROUTE NETWORKS USING USER-DEFINED KEYWORDS

A PROJECT REPORT

Submitted by

DRISHTI CHAUHAN [Reg No: RA1411008010265]
GOPALIKA SHARMA [Reg No: RA1411008010286]
SHUBHAM MEHROTRA [Reg No: RA1411008010341]

Under the guidance of

Ms. K. SORNALAKSHMI

(Assistant Professor Sr.G, Department of Information Technology)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Kancheepuram District

April 2018

SRM Institute of Science and Technology

(Deemed to be University u/s 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled "**MANAGING ROAD ROUTE NETWORKS USING USER-DEFINED KEYWORDS**" is the bonafide work of " **DRISHTI CHAUHAN [Reg No: RA1411008010265], GOPALIKA SHARMA [Reg No: RA1411008010286], SHUBHAM MEHROTRA [Reg No: RA1411008010341]**"who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Ms. K. SORNALAKSHMI
GUIDE
Assistant Professor Sr.G
Dept. of Information Technology

SIGNATURE

Dr. G. VADIVU
HEAD OF THE DEPARTMENT
Dept. of Information Technology

Signature of the Internal Examiner

Signature of the External Examiner

ABSTRACT

As our era is advancing towards the growth of geo-positioning technologies, people are in dire need of a personalized route query. In the recent years, work has been done towards an ideal course covering an arrangement of watchwords. Although, an optimal route might not always be the road one wants to go through. A personalized query can be defined by a few signs that portray the spatial setting between Points of Interest along the course which can immensely differ in comparison to the optimally defined route. Therefore, we have broken down the clue based research algorithm which enables the user to mention their own keywords and establishes their inter structural association. First, we propose a greedy algorithm and then we move on to a dynamic programming algorithm. These algorithms form the headlines of the aforesaid idea. For improving the efficiency of our algorithm we take the help of Branch and Bound algorithm which is known to cut down the extra and unnecessary details which might be present in any of the vertices. In order to accelerate the searching process , we introduce an AB-tree that stores both the computed separate and the client entered catchphrase in the tree structure. In order to diminish the index value we develop a PB-tree by using the jump name file esteem to land at the pinned location of the user. Extensive and hard-geared experiments are conducted which verify our algorithms.

DECLARATION

We hereby declare that the Major Project titled **Managing road route networks using user-defined keywords** to be submitted for the degree of Bachelor of Technology is my original work, and the dissertation has not formed the basis for the award of any degree, diploma, association, or fellowship of similar other titles. It has not been submitted to any other university or institution for the award of any degree or diploma.

Drishti Chauhan

Gopalika Sharma

Shubham Mehrotra

ACKNOWLEDGEMENTS

We would like to express our heartfelt gratitude and regards to **Dr. G. Vadivu**, Head of the Department of Information Technology, for providing us with this opportunity to present our project on **Managing road route networks using user-defined keywords**. We Would also like to convey our heartiest thanks to our Project Guide **Ms. K. Sornalakshmi** and our Project Coordinator **Ms. M. Maragatham** for their valuable guidance, consistent encouragement, personal caring, timely help, and for providing us with an excellent atmosphere for doing this project. All through the work, in spite of their busy schedule, they extended cheerful and cordial support ti us for completing this project. We give our heartiest thanks to all the faculty of the Department of Information Technology and friends for their valuable advice and encouragement.

Drishti Chauhan

Gopalika Sharma

Shubham Mehrotra

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

CRS Clue-based Route Search

POI Points of Interest

GCS Greedy clue search

CDP CLue-based Dynamic Programming

CHAPTER 1

1.1 INTRODUCTION

With the fast advancement of area based administrations and geo-situating advances, there is a reasonable inclination to make more geo literary articles accessible in numerous applications. For instance, area data and compact brief depictions of a few organizations (eg eateries, lodgings) can without much of a stretch be found in online neighborhood look administrations (eg, business directory). To give a superior client encounter, a few spatial questioning methods and catchphrases related models have risen, with the goal that geo-textiles can be effectively recovered. It is normal to discover a state of intrigue (PoI) by giving a correct address or a discernible catchphrase (that is, just a couple of Points of Interest (POI) contain the particular watchword) in an area that can extraordinarily distinguish the position. Some current employments stretch out this inquiry to more complex setups, for example, recouping a gathering of geo material items (ordinarily more than 2) or a direction that incorporates various catchphrases.

Dissimilar to your activity, we will probably locate a practical arrangement course in street organizes using tracks. Specifically, in this archive, we have examined another sort of inquiry, ie a look for courses in view of signs Clue-based Route Search (CRS), which enables the client to give pieces of information about content and space setting along the way such that a superior coordinating way with the beginning stage the tracks are returned. All the more particularly, a CRS inquiry is characterized on a G street system, and section to the question comprises of a source vertex vq and an arrangement of follows, in which each follow contains a watchword question and a client gave organize remove. A summit contains a watchword key is considered as a vertex of correspondence. The inquiry restores a way P in G that begins in vq, with the goal that passes (I) P through a succession of comparing vertices (Poi) w.r.t. the signs and (ii) the system

removes between two relating infectious vertices they are near the separation indicated by the comparing client such that the client's inquiry purpose is fulfilled.

1.2 LITERATURE REVIEW

1.2.1 Application Scenarios

Modelling User Intention:

The user might refer to a particular POI in a different way as it perceived usually.For example he may refer a canteen as a restaurant and this should not compromise the search and the result hence this is taken care of with this logic.

Increased flexibility:

In genuine situations, an ideal course is not required rather a personalized route is preferred. The user may have some requirements while planning a trip. Let us consider a particular scenario, a user wants to find a late night fast food place near a cinema hall where he plans to watch a movie and thus he can pick up his late dinner if it is at a walking distance from the cinema hall. These customized necessities influence the course to seek more separation touchy and more adaptable with the end goal that the separation between the POIs along the course ought to be as close as indicated by the client.

Clue based navigation:

Usually the drivers come across such instances when they have to ask for help and they get quite a detailed guiding from the people instructing them. As in, go straight on the road for 200 meters and then turn left from a temple and then turn right at the next crossing and you will reach your destination. In this way, a novel kind of course seek which consequently deciphers the pieces of information contained in such answers happen to indispensable significance. On the off chance that we incorporate this on our

present route benefits, a superior client experience can be guaranteed.

1.2.2 Overview of Data Mining

Data mining harbours sophisticated data analysis tools to discover unknown, valid patterns and relationships in large data sets. These tools can comprise of statistical models, mathematical algorithms, neural networks or decision trees. Data mining is not just collecting and managing data, It is also analyzing the data and predicting the required.

The main purpose of data mining is to find certain new, potential and important, and linking relations to each other and forming a pattern of the existing data. Information extraction, data revelation, data reaping, information paleontology and information design handling set up together are utilized to discover appropriate examples and information.

Data mining as a term is importantly is used by database researchers, and statistics using researchers and the communities of business people. The term Knowledge Discovery Derivable refers to the process of discovering knowledgeable information from the data, where data mining is integral to it. The process involved in KDD, such as preparing the data and selecting while cleaning and interpreting it to ensure that useful knowledge is derived from the data.Data mining is an extensive approach of analyzing of data in a traditional manner using statistics as it harbours approaches and derivatives from various disciplines, to name a few like AI, IOT, OLAP etc.

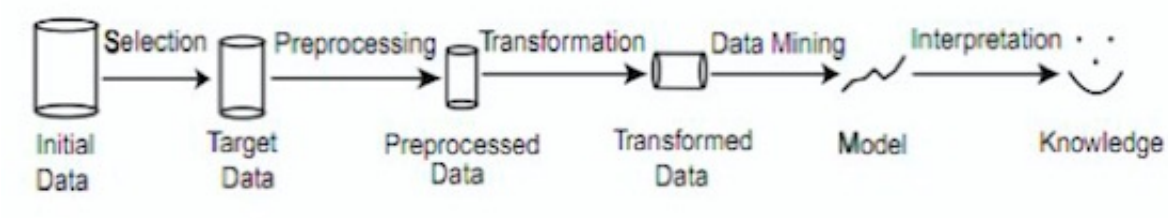

Figure 1.1: Knowledge Discovery Process

The datasets that we stored in our our virtual database went through the following series of adjustments before getting the final result, in order to calculate the highest probablity the following are the correct step to go through. The information mining

offers different capacities and a fluctuated feeling of learning, they are portrayed as under:

Characterization: At the point when the general highlights of a question in an objective class are portrayed, they create trademark rules. Typically a database question recovers the information in view of the pertinence to a client indicated class and gone through a synopsis module to separate the substance of the information at various levels of reflections. For instance, one may wish to portray the clients of a store who frequently lease in excess of 30 motion pictures a year. The property situated acceptance technique can be utilized to complete information rundown considering the idea progressive systems on the traits that portray the objective class. With an information 3D shape that contains the report of information, straightforward OLAP activities fit the motivation behind information portrayal.

Discrimination: Information separation is essentially the correlation of the general highlights of articles between two classes alluded to as the objective class and the differentiating class and hence creates the discriminant rules. For instance, a distinction can be noted in the qualities of a client who leased in excess of 30 motion pictures in the most recent year with those whose rental record is lower than 5. The strategies utilized here are like the methods utilized for information portrayal with the special case that information segregation comes about incorporate relative measures.

Association analysis: Affiliation examination as a rule distinguishes the things that happen more frequently(frequent thing sets) in a value-based database in view of an edge called bolster. Another edge, certainty, which is the contingent likelihood that a thing shows up in an exchange when another thing shows up, is utilized to pinpoint affiliation rules. Market container investigation is typically done by this. For instance, a supervisor of a store can think that its valuable to comprehend what motion pictures are leased regularly, together or if there is some connection between leasing a specific sort of films and purchasing popcorn or pop. The found affiliation rules are of the shape: P.Q [s, c], where P and Q speak to the conjunctions of characteristic esteem sets, and s (bolster) is the likelihood of event of P and Q together in an exchange and c (certainty) is the restrictive likelihood that Q shows up in an exchange when P is available.

Classification: Characterization sorts out the information in given classes, and uses

given class marks to arrange the articles in the information gathering. The methodologies that are utilized as a part of order utilize a preparation set, in which objects are as of now appointed to various class marks. The grouping calculation gains from the preparation set and constructs a model. The model is utilized to arrange new questions. For instance, once a credit arrangement is begun, the supervisor can examine the clients conduct versus their Mastercards and in this manner name them appropriately with three conceivable levels, to be specific, protected, hazardous, extremely unsafe . The arrangement examination helps in producing a model that could be utilized to either acknowledge or dismiss credit asks for later on.

Prediction: Expectation has given the potential ramifications of effective guaging in a business setting and has increased extensive consideration. Expectation has two noteworthy orders: one can either endeavor to anticipate some inaccessible information esteems or pending patterns, or foresee a class mark for a few information. The last is fixing to arrangement. On the fruition of a grouping model, in view of a preparation set, the class name of a question can be predicted in light of the characteristic estimations of the protest and the quality estimations of the classes. Expectation is regularly alluded to the figure of increment/diminish slants in time related information. The significant thought behind forecast is to the utilization of extensive number of past qualities to think about likely future esteems.

Clustering: Grouping is like characterization, in the association of information in classes. The class names are obscure in grouping and it is up to the bunching calculation to find worthy classes dissimilar to order in which the class marks are known. Since grouping isn't managed by given class names, bunching has additionally been named as unsupervised arrangement. There are a few bunching approaches, yet they all depend on just a single rule, either the comparability between the objects of same class can be maximised(intra-class likeness) or the closeness between objects of various classes would b be able to limited (between class similitude).

Outlier analysis: There are a few information components that can't be assembled in a given class or group, and are consequently named as anomalies. Despite the fact that they are considered as exemptions or astonishments, it is essential to recognize them. In the greater part of the applications they are considered as commotion and are

disposed of, yet they can uncover vital learning in different areas, and consequently be extremely huge and their examination is profitable.

Evolution and deviation analysis: The investigation of the information that progressions with time is advancement and deviation examination. Advancement investigation speaks to the patterns in information, which are as per the describing, looking at, ordering or bunching of time related information. Deviation examination manages the contrast between the deliberate esteems and expected esteems and finds the reason for deviation from the foreseen esteems.

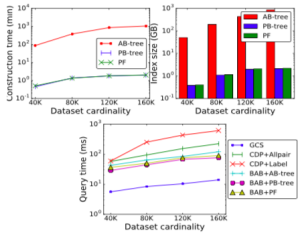

Fig. 8. Effect of the dataset cardinality

Figure 1.2: Dataset cardinality

CHAPTER 2

2.1 SYSTEM ANALYSIS

2.1.1 Overview

Existing system: Greedy clue-based algorithm GCS with no index where the network expansion approach is adopted to greedily select the current best candidates to construct feasible paths. Devise an exact algorithm, namely clue-based dynamic programming CDP,to answer the query that lists every feasible path and finally gives the most optimal result. BAB can be implemented by filter and refining the paradigm. so that a limited amount of veritces are visited, in order to improve the efficiency.

Disadvantage: Greedy calculations don't work for a few issues. In spite of their straightforwardness, redress avaricious calculations can be inconspicuous. It's anything but difficult to trick yourself into trusting an inaccurate eager calculation is right. "I can't think about a counter-case, so there are none." Utilizing greediness isn't a programmed.

Proposed system: Clients may offer notice to a model more suited their needs with POI evaluations, similar to normal cost of an inn room and so on in the question. It is important to consider worldly data and further expand the CRS inquiry. Every POI is relegated with an opening hours interim and each piece of information has a meeting time t, where the chose inquiry needs to discover a way so the interim time coordinates the interim POI covering the meeting time. Expecting clients to give correct catchphrase coordinate is hard now and then for they are profiting an intimation which may be not up to the mark.So, it is our basic to make the model sufficient to help the watchword entered by the client roughly.

Advantage: Useful in job shop scheduling, automatic programming, circuit designing, and vehicle routing and portfolio management. It is also helpful to solve pure

optimization problems where the objective is to find the best state according to the objective function. It requires much less conditions than other search techniques.

2.1.2 System Specifications

Hardware specifications:

System : Pentium IV 2.5 GHz. Hard Disk : 44 GB. Monitor : 21 VGA Colour. Mouse : Logitech. Ram : 512 Mb.

Software specifications:

Operating system : Windows 8. Coding Language : Java Tool : Netbeans 8.1 Database : MYSQL SERVER 2008, phpmyadmin.

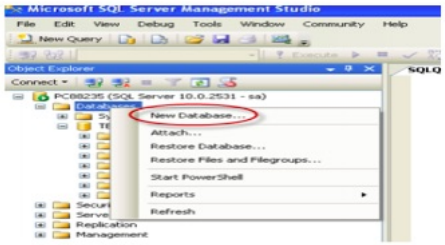

Figure 2.1: Environment

CHAPTER 3

3.1 ALGORITHMS IN PLAY

3.1.1 Greedy clue search algorithm:

A greedy calculation is an algorithmic worldview that takes after the critical thinking tenet of settling on the best decision in the region at each phase with the expectation of finding a worldwide ideal. In numerous issues, an avaricious system does not give the best arrangement but rather when all is said in done, it furnishes us with an answer that is shut in closeness to the all inclusive ideal arrangement inside a less time.

For example, a strategy for the traveling salesman problem (which has a high number crunching process and thus a high complexity), using the greedy algorithm, follows the following rule: "Visit a city that is closest to the current city and has been unvisited, in each stage". This strategy might not provide the best solution, but will remove a certain number of steps, and thus provide a solution that has some unnecessary steps. In numerical improvement, eager calculations take care of the integrative issues having the properties of frameworks. Greedy algorithms fail to find the globally optimal solution in most of the cases, but not always, as they do not operate thoroughly on all the data. They often react at very early stages and at certain choices which prevents them from finding the best overall solution later. For example, all known greedy algorithms used for the problems of graph coloring and all the other NP-complete problems do not find the optimum solutions readily. However, they are valuable as they respond rapidly and often result in ideal estimates.

If a greedy algorithm provides the best solution world- wide for a certain problem class, then it typically becomes the method of choice unlike dynamic programming, as it is faster than those optimization methods. The examples of greedy algorithms for finding the minimum spanning trees are Kruskals and Prims algorithm, and the algorithm to

find optimum Huffman trees. The classes of such algorithm is provided by the theory of matroids and the theory of greedoids.

Greedy algorithms are used in network routing as well. With the help of greedy routing, a message or a keyword is forwarded to the adjacent node, which is closest to the destination. The impression of a nodes location (and therefore "closeness") may be determined by its physical location, as in we can use geographic routing that is used by ad hoc networks. Location used might entirely be a simulated construct as in small world routing and the distributed hash table.

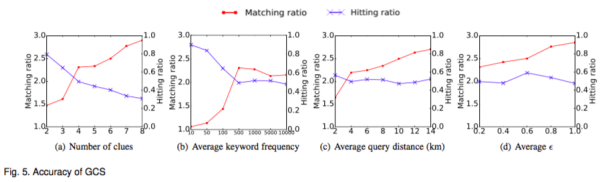

Figure 3.1: Greedy clue search

3.1.2 Clue-based dynamic program algorithm

As we know, Greedy clue search (GCS) responds quickly than any other algorithm yet the accuracy of the answer is sometimes compromised. In order to achieve better accuracy while answering the CRS query, we propose a calculation, called Clue-based Dynamic Programming (CDP). For the most part, it is trying to build up a calculation for the CRS inquiries, since the thoroughgoing procedure to scan for POIs in the street systems can't be maintained a strategic distance from .

In CLue-based Dynamic Programming (CDP), we make a watchword position list for every catchphrase w(the piece of information that the client enters), which is a rundown of vertices that contain w. At whatever point a CRS question is created, the posting records are arranged by the catchphrase . Note that the request of the posting rundown can be self-assertive as it doesn't generally make a difference, and consequently are arranged by vertex id for straightforwardness. It is effortlessly observable that a k-bipartite diagram G has been built by these posting records, which thus demonstrates every conceivable way for a given C. The heaviness of each edge in G is registered as the coordinating separation. In particular, for every u, we characterize D(wi,u) that means the base coordinating separation which can be accomplished with a walk that

passes the watchwords from w1 to wi predictable with the request in C and stops at u. As it were, the heaviness of vertex u G is processed by D(wi,u), which is the base coordinating separation of all mostly possible ways end at u.

3.1.3 Branch and Bound algorithm

Branch and bound (BB, BB, or BnB) is a productive system for taking care of discrete and combinative streamlining issues, and additionally scientific improvement. A branch and-bound calculation comprises of an orderly and sorted out rundown of competitor arrangements by methods for state space seek: the arrangement of the hopeful arrangements shapes an established tree with the full set at the root. The calculation at that point investigates the branches of this tree, which are essentially the subsets of the arrangement set. Now the branch is created and for further listing out the solutions of this branch, it is first checked against the upper and lower bounds on the feasible solution. The solution is kept if it produces the best result, and is discarded if it does not provide a better solution when compared to the best solution that has been generated by the algorithm so far. The calculation is reliant on the branch of the pursuit space or the productive guess of the upper and lower limits of the hunt district and plays out an exhaustive identification as the size (n-dimensional volume) of the locale approaches zero. The technique was first proposed by A. H. Land and A. G. Doig in 1960 for discrete programming, and has turned into the most ordinarily utilized instrument for settling NP-hard improvement problems.The name "branch and bound" first happened in crafted by Little et al.on the travelling salesman problem.

Overview:

The goal of a branch-and-bound calculation is to discover an esteem x that will either amplify or limit the estimation of a genuine esteemed capacity f(x). This genuine esteemed capacity is called a goal work, among some set S of acceptable, or competitor arrangements.The set S is known as the pursuit space, or achievable area.It is assumed by the rest of the section that minimization of f(x) is wanted; there is no loss in gener-

alization while keeping the assumption, since the maximum value of f(x) can be found by the minimum of g(x) = f(x). A BB algorithm functions according to two principles: The search space that has been acquired is additionally partitioned into littler spaces, consequently limiting f(x) on these littler spaces. The part performed is called fanning. Fanning alone adds up to beast drive posting of hopeful arrangements and testing them all. To upgrade the execution of animal power seek, a BB calculation keeps up a record of the limits on the base that it is endeavoring to discover, and utilizes these limits to "prune" the pursuit space, along these lines expelling the applicant arrangements that it can demonstrate will not contain the best arrangement. These principles can be turned into concrete algorithms by use of some kind of data structures that represent data sets of candidate solutions for specific optimization problems. The representation of these data sets by data structures is called an instance of the problem. The set of candidate solutions of an instance I is denoted by SI.

The instance representation comes with three operations: Branch(I) produces two or more instances, each representing a subset of SI. (To prevent the algorithm from visiting the same candidate solution twice, these subsets are used, but this is not required. For a correct BB algorithm, the only thing that required is that the best/optimal solution among SI should be present in at least one of the subsets). The lower bound on the value of any candidate solution in the space represented by I is computed by Bound(I), that is, bound(I) and f(x) for all x in SI. Solution(I) helps in determining whether I represents a single candidate solution. (if solution fails to do so, the operation may return some feasible solution from among SI.) With the help of these activities, a BB calculation looks recursively in a best down manner, through the tree of cases, shaped by the branch activity. On going to a case I, it checks whether bound(I) is more prominent than the upper destined for some other occurrence that it as of now went by; assuming this is the case, I might be securely disposed of from the inquiry and the recursion stops. A global variable records the minimum upper bound that has been seen so far, and thus pruning is implemented.

3.1.4 Binary Tree

In the field of computers, a binary tree is a data structure shaped like a tree in which almost each node has two children attached to them, which are known as the left child and the right child according to their layout. According to a recursive definition, with the help of a set theory one can say that a non-empty tree is a tuple and an empty set is singleton set. A few authors consider if the binary tree is also an empty set.

Drawing a perspective from a graph theory binary trees can be defined as actual tree-like looking structure. Binary tree may also be known as a parting tree-like looking structure which term was used in pretty old books of computer programming, before the new era of computers dawned on us.It is easy to deter that a binary tree has no sense of direction rather than a directed one, in which the binary tree will be considered an ordered and rooted. Some authors pull in these two terms together rather than submitting two separate opinions regarding rooted or unrooted binary tree, but it is assumed that a binary tree is always rooted. A special case of an ordered K-ary tree with the value of k being 2 is a binary tree.

While computing, binary tree structure is not given much attention or heed. A tedious task is when we want to label the nodes, where every node should hold on to some value. When we label the nodes in this particular way, we call them either search trees or heaps, and are used for efficient sorting and searching. When we have left or right nodes with no parent node or a root node, we need to look after the designation as many applications are based on it. When it comes to the stream of mathematics, what is known as binary tree can flunctuate from one author to the other. Some prefer the computer science definition, but others just call it a node with no children whatsoever and if having children but not arranged.

Types of binary tree that have been used for this application;

All pair binary tree

Pivot based reversal binary tree

CHAPTER 4

4.1 SQL SERVER–phpmyadmin

4.1.1 Introduction

Microsoft is the distributor of SQL server. There are different editions of SQL Server which are available in the market, where SQL Server Express is available to the customers for free, they can simply download and use it.SQL Server uses T-SQL(Transact-SQL). T-SQL is an extension to the property of SQL. TSQL and standard SQL have similarities but in addition T-SQL has some extra functions like built in functions.
T-SQL, adds onto the SQL with various functions for processing of string, data, mathematics etc. SQL Server comprises of Database engine and a Management studio. The engine is devoid of graphical interface- it just acts a service running in the background. The management studio helps the user to view the data using graphical tool. It can be installed both on the server and the client. SQL management studio can be used to create a novel database, by bringing any change possible by modifying tables and indexes. It takes in the queries through the windows which provide an interface similar to a GUI based interface, which is used to write and run queries.

4.1.2 Create a new Database

It is extremely simple to create a new database in Microsoft SQL server. Just use the mouse to right click on the database node and select a new data base from the drop down menu.
The OLAP administrations highlight accessible in SQL server rendition 7.0 is called SQL server 2000 examination administrations. The term OLAP administrations is currently called Analysis administrations. It likewise has another information mining segment in play. The vault segment accessible in SQL server form 7 is presently known

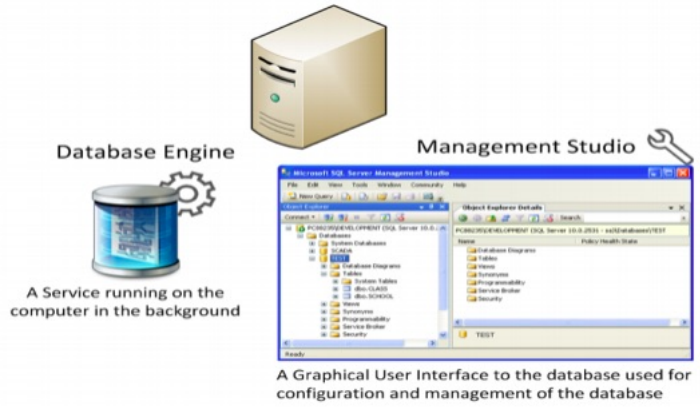

Figure 4.1: Database Studio

as microsoft SQL server 2000 meta information administrations. The term repository is used only in the reference to the engine within meta data services; consisting of six types of objects, They are,

1. TABLE

2.QUERY

3. FORM

4. REPORT

5.MACRO

We are going to cover the first two as that is of our importance.

Table

A database is a collection of data supporting a specific topic. A table view that we can work with are of two types, 1. Design View 2. Datasheet View design In order to modify the structure or the layout o the table we use the design view. We can specify the type of data that one can hold in the table. In order to add, edit or bring about any kind of analysis on the data itself, we work on the view mode.

Query

A query is a doubt that has to be asked to the data. The particular data is accessed which which answers the question asked, from one or more table. The data answering to our question is either a dynaset or known as a snapshot. Each time after we run a query, we get to store the information in the dynaset. The access is either used to display the dynaset or snapshot for customers to have a view or perform an action such as deleting

or bringing about an update.

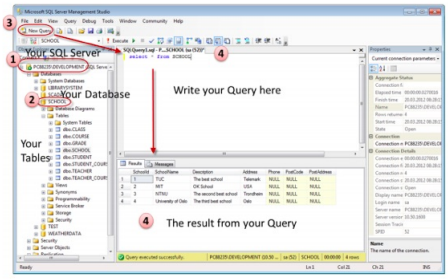

Figure 4.2: query

4.1.3 Need of a SOL server

Favorable circumstances of SQL SERVER over others:

More secure: Get layers of protection for your data still and in development. SQL Server is evaluated the smallest powerless database six years continuously by the National Institute of Standards and Technology.

End-to-end compact BI on any contraption: Disseminate current reports using Power BI or Excel 2016 to an iPhone, Android, or Windows phone. Engage both self-advantage BI and versatility at a little measure of the cost of Oracle.

In-database advanced examination at no extra cost: Settle on better decisions with easy to-use examination instruments that reliably unite undertaking data, external data, and unstructured tremendous data.

In-memory worked in finished all workloads: Achieve exponential execution gets with the speed of in-memory online trade planning.

Dependable experience from on-premises to cloud: Get an anticipated experience both on-premises and in the cloud with general headway and organization instruments and essential T-SQL.

phpmyadmin–purpose

Advantages of phpMyAdmin with SQL Server:

a. Commonly introduced on oversaw facilitating situations.

b. Web Based which implies you can access from any PC.

c. Local assets aren't utilized while interfacing.

d. Simplicity.

e. Has a UI and you can run queries inside the SQL.

f .Can paste queries into the SQL to test data output; from a simple Select FROM tablename to more advanced relational queries using various tables.

g. Import And Export of .sql file is present.

h. The copying and pasting of queries is available with the phpMyAdmin console.

i. Remote Access: Directly Connect With Data on Your Server.

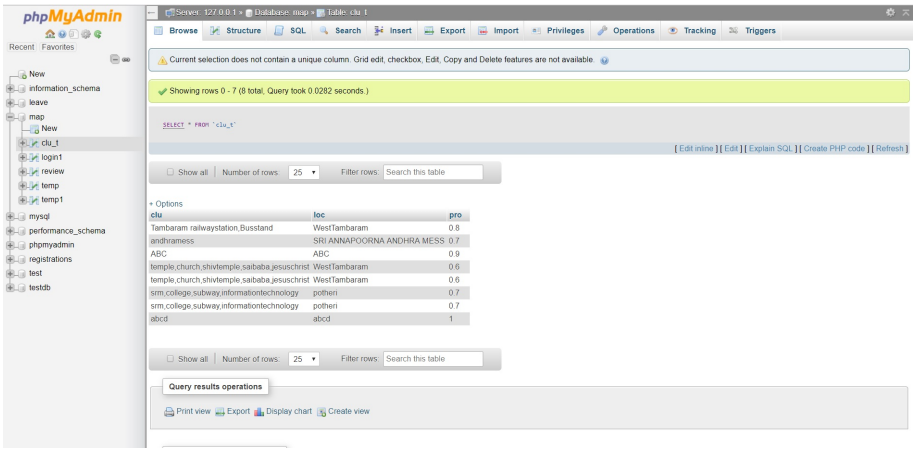

Figure 4.3: Storing data in database

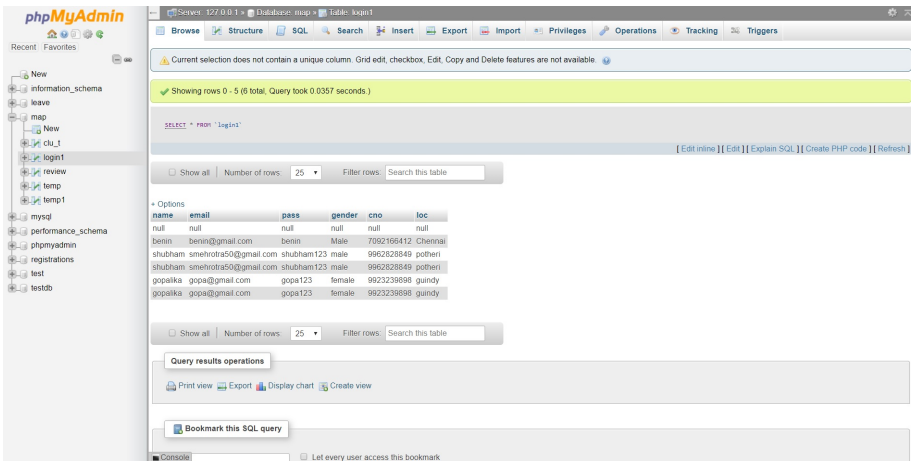

Figure 4.4: Data in database

CHAPTER 5

5.1 System Testing

5.1.1 Testing Objectives

Testing is the process in which we discover errors and faults in the final product. The final workproduct is checked whether it performs all the functions that it was supposed to perform in the earlier analysing and designing phase, and that it satisfies all the user specifications without any fail.Testing has various forms which require different input and output and thus check different parts of the product in different stages.

5.2 Types of tests

5.2.1 Unit Testing

Unit testing is done by executing the test cases on the individual units of the source code. The goal is to check whether the internal program logic functions properly and hence produces valid outputs.This is performed after the completion of individual units of the program before integration, thus validating all the decision branches and internal code flow. This is a structural testing, which performs basic tests at the component level and tests specific business process, application or system configuration and thus relies on knowledge of its manufacture. The goal is to isolate each part of the program and check whether the individual parts perform accurately according to the documented specifications and has clearly defined inputs and expected results.

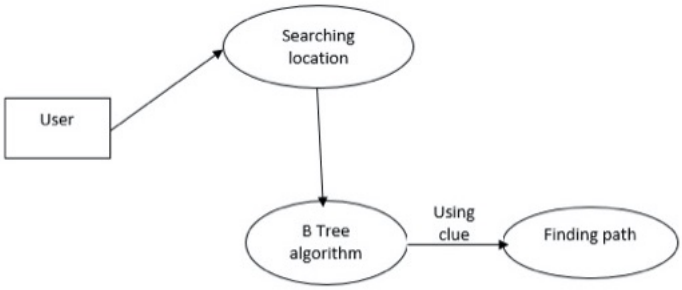

Figure 5.1: Processing

Test Strategy and Approach

Field testing is carried out manually. Test Objectives All the inputs that are given must work properly. The links that have been provided for certain pages must work and should be active when released There should be no delay in the process of sending and receiving texts, in the opening of the screens Features to be tested The format of the inputs should be correct as mentioned in the manual. There should be no double entries for one input The link should take user to the page as per requested.

5.2.2 Test cases on the application

Positive Test cases:

Test ID	Positive Test Cases	Expected Result	Actual Result	Status
1	Open Map Application.	The app should properly open.	The app opened properly.	Pass
2	Enter Origin Address.	The source textfield should take the origin address.	The source textfield took the origin address.	Pass
3	Enter Destination Address.	The destination textfield should take the origin address.	The destination textfield took the origin address.	Pass
4	Click on 'FIND PATH'.	The button should work properly.	The button worked properly.	Pass
5	Check the distance is right.	The app should show the right distance in kms.	The app showed the right distance in kms.	Pass
6	Check the time required to reach the destination is right.	The app should show the right time.	The app showed the right time.	Pass
7	Click on 'A' to check the source is right.	The pinpoint A should show correct source on map.	The pinpoint A showed correct source on map.	Pass
8	Click on 'B' to check the destination is right.	The pinpoint B should show correct destination on map.	The pinpoint B showed correct destination on map.	Pass
9	Zoom in for clear view.	The map should be able to zoom.	The map zoomed.	Pass
10	Click on 'Google Maps' Button to see it in Google Maps.	The Button should work properly, open Google Maps and show the location.	The Button worked properly, opened Google Maps and showed the location.	Pass
11	Click on 'Directions' Button to see it in Google Maps.	The Button should work properly, open Google Maps and show direction.	The Button worked properly, opened Google Maps and showed direction.	Pass

Figure 5.2: Positive Test cases

Negative Test Cases :

Test ID	Negative Test Cases	Expected Result	Actual Result	Status
1	Type wrong source name in Map Application.	The app should not show anything.	The app showed nothing.	Pass
2	Type wrong destination name in Map Application.	The app should not show anything.	The app showed nothing.	Pass
3	Type wrong name of both source and destination in respective text boxes and click 'FIND PATH'.	The app should not show anything.	The app showed nothing.	Pass
4	Click 'FIND PATH' and type source and leave destination blank.	The app should not show anything.	The app showed nothing.	Pass
5	Click 'FIND PATH' and type destination and leave source blank.	The app should not show anything.	The app showed nothing.	Pass
6	Type source in destination text box and destination in source text box and click 'FIND PATH'.	The app should not show anything.	The app showed nothing.	Pass
7	Type source of one country and destination of other country and click 'FIND PATH'.	The app should not show anything.	The app showed nothing.	Pass

Figure 5.3: Negative Test cases

CHAPTER 6

6.1 EXPERIMENTS

6.1.1 Experimental Settings

All these algorithms introduced in this report were implemented on Netbeans platform usinf java language on windows and run on Intel with a 32GB RAM.

Datasets:

Real datasets were used, the road network of various places in Chennai like Potheri, T nagar , Tambaram etc. The dataset contains an undirected weighted diagram that is a piece of the street arrange. The heaviness of each edge in a chart speaks to the separation between the two vertices of the edge.

Parameter settings:

In order to make our algorithms run, we generatean arbitrary of 100 questions for each arrangement of trial and hence measure their execution by assessing their normal. In order to make the algorithms work in an efficient way we make the value of parameters vary a little. For default settings we choose 16k as the number of vertices; dataset cardinality, 4 is taken as the number of clues and 64 stands for hash code length. Here, we are assuming that a keyword at most shows up one time at a vertex, thus the frequency of a keyword w is the number of vertices that contain w. According to a statistical approach, the frequency of keyword shows the percentage of keywords with varied frequencies. In the inquiry, the frequencies of watchword is haphazardly produced with the normal separations and the certainty factors.

Performance evaluation:

The performance comparison of all the algorithms are done with the respect of query processing, record size and development time. The hop label comparison is excluded from our evaluation as of now. In order to venture into query time processing and evalu-

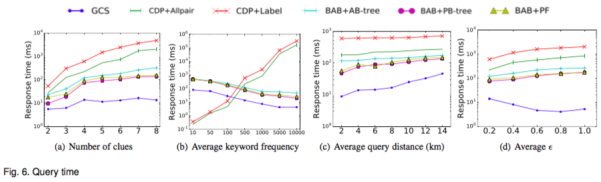

Fig. 6. Query time

Figure 6.1: Query time processing

ation, it is easy to find BAB performing better than GCS and CDP. Besides, applyingall-match in CDP has a far shorter reaction time yet cost extensive costing than using jump name, and utilizing PB tree in BAB has a superior execution than when one uses AB tree and PF. To consider, the truth of the matter is that for file size and development time, mark based methodologies have genuinely substantially littler size and less time than all match based methodologies.

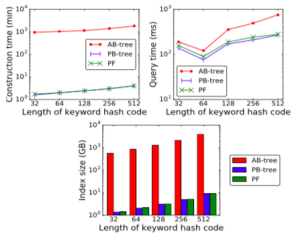

Fig. 7. Effect of the keyword hash code length h

Figure 6.2: Hash effect

6.2 Practical implementations

6.2.1 The "website"

The sole purpose of this website is to process all the clues entered by an individual and provide him with the most optimal route, it gives various modes of transportation as an option like walking, driving etc.There is one user review page where one can write their reviews. It is easy to set up an account by simply signing up. The prototype look of the website can be witnessed as under.

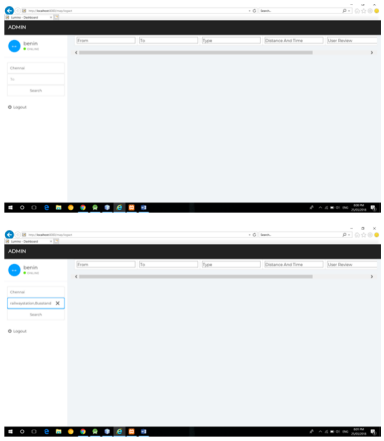

Figure 6.3: Entering keywords

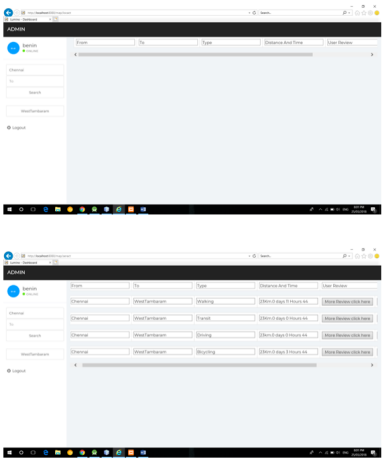

Figure 6.4: Modes of Transportation

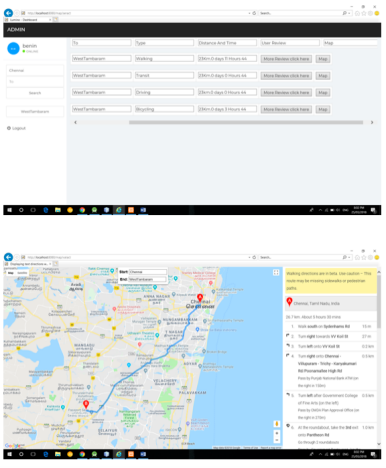

Figure 6.5: Map with directions

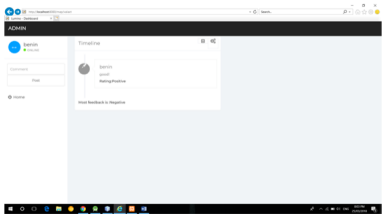

Figure 6.6: Review page

6.2.2 The "app"

An android app was also introduced by us for the same purpose as in this technological advanced world people prefer apps for everything so we tried to put it all together in an android application.The work on the application is ongoing.

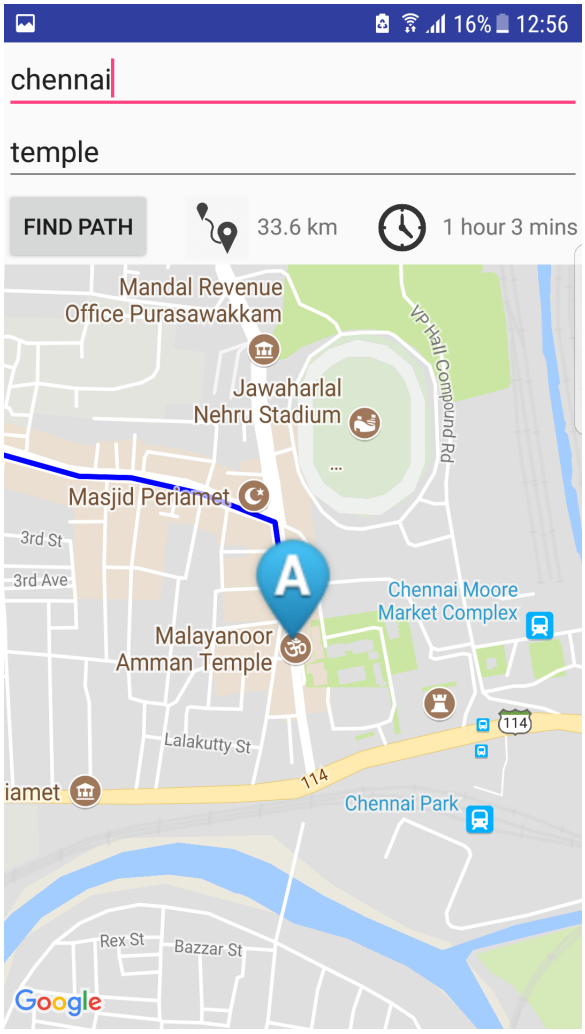

Figure 6.7: Working of the app

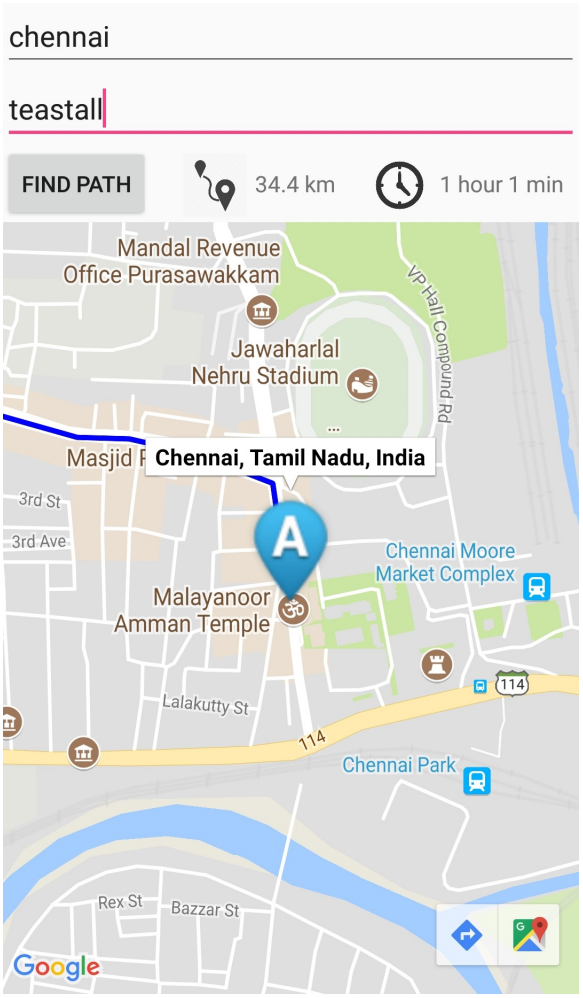

Figure 6.8: User defined Key words

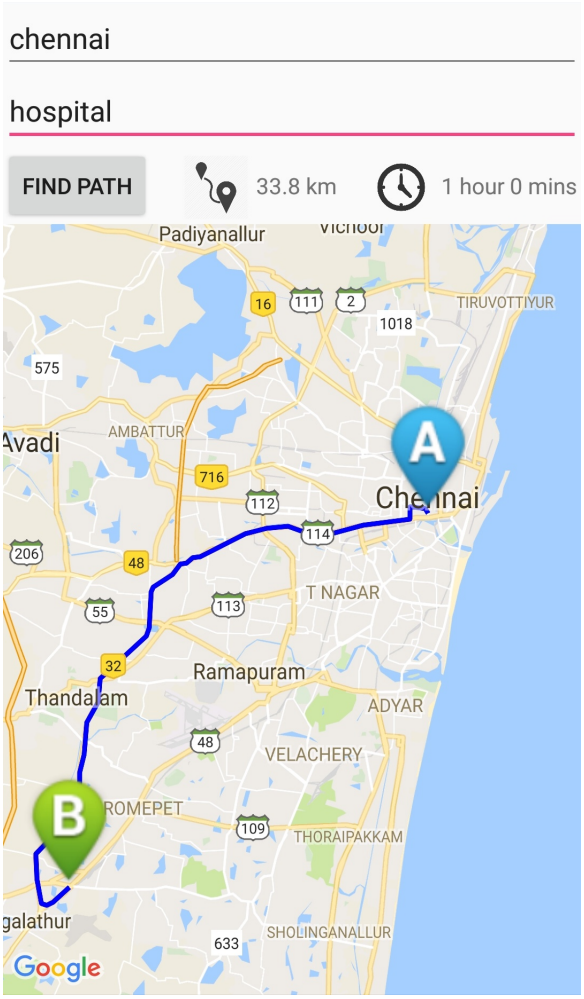

Figure 6.9: Implementation of the keyword

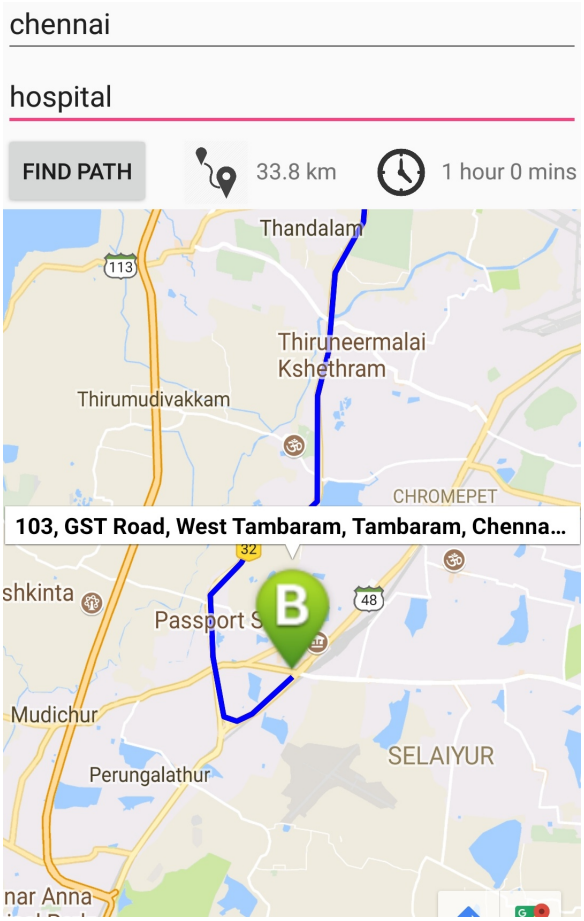

Figure 6.10: Defining the location

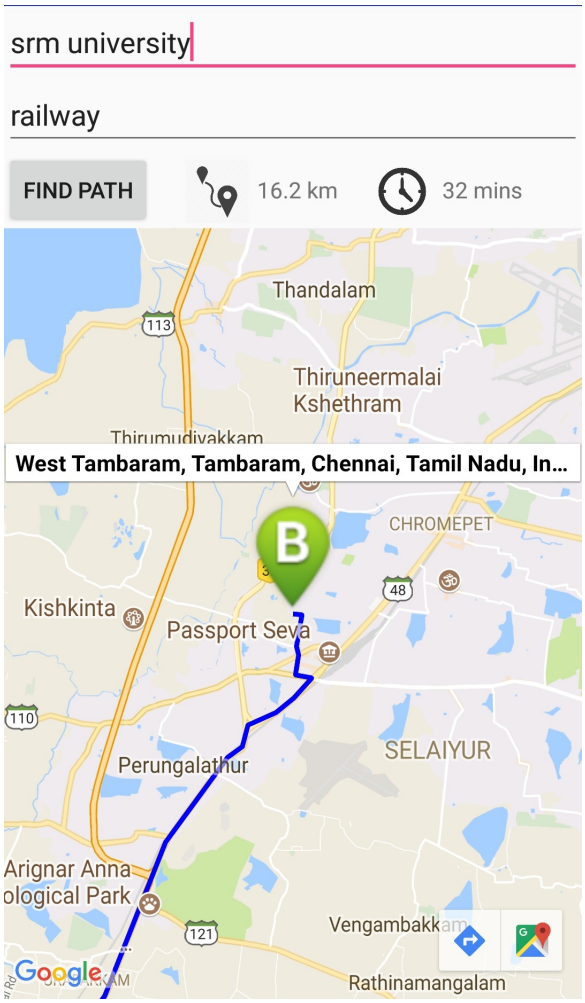

Figure 6.11: Example

6.3 Related Work

6.3.1 Travel Route Search

The travel route search has been an important example and a substantial study for a number of decades.Travelling Salesman Problem(TSP) is the classiest problem when it comes to route planning. TSP aims in finding the most cost effective and efficient route from a source point to a set of targets. Like studying the problem of Trip Planning Query(TPQ) in spacial database, where very object has a location and a category. Let us take the starting point as S, a destination as E and a set of categories as C, TPQ tries to pull off a path which travels from all the categories at least once before reaching E as the destination. TPQ can be considered as the generalisation of travelling salesman problem, thus two approximate algorithms are proposed. Studies in optimal sequenced route (OSR) aims to find an optimal path which runs through all the typed vertices, reaching the required destination after having covered the specific sequence imposed on the types of locations. At that point appears the LORD or R-LORD calculations to sift through the areas which don't regard fit as areas in the ideal course, along these lines this enhances our inquiry proficiency. The issue in investigations of multi-control fractional succession course (MRPSR) has been recognized. This intends to locate an ideal course with least separation under a lead which underpins incomplete class arrange characterized in the question. They propose three heuristic calculations to look for most extreme ideal answers for the client given inquiry. This from now on brings forth the possibility of an insatiable calculation which finds a course whose length is littler than a predefined edge though the content importance is dependent upon it's greatest.

The issue of finding a way which touches no less than one fulfilling substance of each kind is unquestionably an intelligent approach.In each progression, a client should give it's criticism regarding whether he was fulfilled by the element or not. Consequently, the issue of multi-watchword directing inquiry in a surmised way happens which supplements the standard method for finding the briefest way joined by different catchphrases and a string closeness work. For every watchword, the coordinating point should have an adjusted separation littler than the said limit. It characterizes the issue

of watchword mindful ideal course question, which is to discover a course which covers every single client indicated catchphrase, a particular spending plan and the target score ought to be contemplated. It proposes two unique arrangements, to be specific in reverse and forward inquiries, keeping in mind the end goal to manage the general and the most ideal course inquiry with an aggregate request. It acquires light the issue of customized trip suggestion, which means to locate the most positive outing that quickens the client encounter for a stipulated time and a spending requirement and furthermore takes the vulnerability of voyaging time into play.

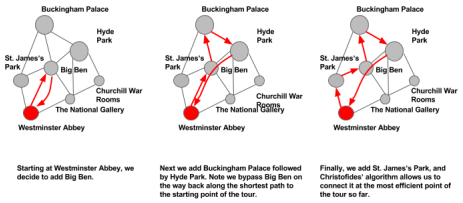

Figure 6.12: Example for Travel route search

CHAPTER 7

7.1 Code Snippets-Android

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context="com.itshareplus.googlemapdemo.MapsActivity"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"  >

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/etOrigin"
        android:hint="Enter origin address" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter destination address"
        android:id="@+id/etDestination" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        >
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Find path"
            android:id="@+id/btnFindPath" />
        <ImageView
            android:layout_marginLeft="20dp"
            android:layout_marginTop="5dp"
            android:layout_width="40dp"
            android:layout_height="40dp"
            android:src="@drawable/ic_distance"/>
        <TextView
            android:layout_marginLeft="5dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="0 km"
            android:id="@+id/tvDistance" />

        <ImageView
            android:layout_marginLeft="20dp"
            android:layout_marginTop="5dp"
            android:layout_width="40dp"
            android:layout_height="40dp"
            android:padding="5dp"
            android:src="@drawable/ic_clock"/>
        <TextView
            android:layout_marginLeft="5dp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="0 min"
            android:id="@+id/tvDuration" />
    </LinearLayout>
```

Figure 7.1: Coding snippets for designing of the app

7.2 Code Snippets-Website

```
package Modules;

import com.google.android.gms.maps.model.LatLng;

import java.util.List;


public class Route {
    public Distance distance;
    public Duration duration;
    public String endAddress;
    public LatLng endLocation;
    public String startAddress;
    public LatLng startLocation;
```

Figure 7.2: Coding snippets for using of maps

```
import android.os.AsyncTask;

import com.google.android.gms.maps.model.LatLng;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;
import java.util.ArrayList;
import java.util.List;


public class DirectionFinder {
    private static final String DIRECTION_URL_API = "https://maps.googleapis.com/maps/api/
directions/json?";
    private static final String GOOGLE_API_KEY = "AIzaSyDnwLP2-WfK8cVZt9OoDYJ9Y8kspXhEHfI";
    private DirectionFinderListener listener;
    private String origin;
    private String destination;

    public DirectionFinder(DirectionFinderListener listener, String origin, String destination)
{
        this.listener = listener;
        this.origin = origin;
        this.destination = destination;
    }

    public void execute() throws UnsupportedEncodingException {
        listener.onDirectionFinderStart();
        new DownloadRawData().execute(createUrl());
    }

    private String createUrl() throws UnsupportedEncodingException {
        String urlOrigin = URLEncoder.encode(origin, "utf-8");
        String urlDestination = URLEncoder.encode(destination, "utf-8");

        return DIRECTION_URL_API + "origin=" + urlOrigin + "&destination=" + urlDestination +
"&key=" + GOOGLE_API_KEY;
    }

    private class DownloadRawData extends AsyncTask<String, Void, String> {

        @Override
        protected String doInBackground(String... params) {
            String link = params[0];
            try {
                URL url = new URL(link);
                InputStream is = url.openConnection().getInputStream();
                StringBuffer buffer = new StringBuffer();
                BufferedReader reader = new BufferedReader(new InputStreamReader(is));
```

Figure 7.3: Coding snippets using google API

4.DirectionFinderListener.java

```
package Modules;

import com.google.android.gms.maps.model.LatLng;

import java.util.ArrayList;
import java.util.List;
```

```
import Modules.Route;


public interface DirectionFinderListener {
    void onDirectionFinderStart();
    void onDirectionFinderSuccess(List<Route> route);
}
```

Figure 7.4: Coding snippets for direction finder

```
@Override
public void onDirectionFinderSuccess(List<Route> routes) {
    progressDialog.dismiss();
    polylinePaths = new ArrayList<>();
    originMarkers = new ArrayList<>();
    destinationMarkers = new ArrayList<>();

    for (Route route : routes) {
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(route.startLocation, 16));
        ((TextView) findViewById(R.id.tvDuration)).setText(route.duration.text);
        ((TextView) findViewById(R.id.tvDistance)).setText(route.distance.text);

        originMarkers.add(mMap.addMarker(new MarkerOptions()
                .icon(BitmapDescriptorFactory.fromResource(R.drawable.start_blue))
                .title(route.startAddress)
                .position(route.startLocation)));
        destinationMarkers.add(mMap.addMarker(new MarkerOptions()
                .icon(BitmapDescriptorFactory.fromResource(R.drawable.end_green))
                .title(route.endAddress)
                .position(route.endLocation)));

        PolylineOptions polylineOptions = new PolylineOptions().
                geodesic(true).
                color(Color.BLUE).
                width(10);

        for (int i = 0; i < route.points.size(); i++)
            polylineOptions.add(route.points.get(i));

        polylinePaths.add(mMap.addPolyline(polylineOptions));
    }
}
```

Figure 7.5: Coding snippets for calculating distance

```
    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        LatLng hcmus = new LatLng(12.825138, 80.045042);
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(hcmus, 18));
        originMarkers.add(mMap.addMarker(new MarkerOptions()
                .title("SRM Tech Park")
                .position(hcmus)));

        if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) !
= PackageManager.PERMISSION_GRANTED && ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
            // TODO: Consider calling
            //    ActivityCompat#requestPermissions
            // here to request the missing permissions, and then overriding
            //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
            //                                          int[] grantResults)
            // to handle the case where the user grants the permission. See the documentation
            // for ActivityCompat#requestPermissions for more details.
            return;
        }
        mMap.setMyLocationEnabled(true);
    }


    @Override
    public void onDirectionFinderStart() {
        progressDialog = ProgressDialog.show(this, "Please wait.",
                "Finding direction..!", true);

        if (originMarkers != null) {
            for (Marker marker : originMarkers) {
                marker.remove();
            }
        }

        if (destinationMarkers != null) {
            for (Marker marker : destinationMarkers) {
                marker.remove();
            }
        }

        if (polylinePaths != null) {
            for (Polyline polyline:polylinePaths ) {
                polyline.remove();
```

Figure 7.6: Coding snippets for marking the destination

```
                                                catch (SQLException ex) {
            Logger.getLogger(locact.class.getName()).log(Level.SEVERE, null, ex);
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(locact.class.getName()).log(Level.SEVERE, null, ex);
        }
            sn.setAttribute("loc2",to);
              rd=request.getRequestDispatcher("Mainpage.jsp");
              rd.forward(request,response);
    }
    }
    else
    {
         try {
             Class.forName("com.mysql.jdbc.Driver");

        con = DriverManager.getConnection("jdbc:mysql://localhost:3306/map","root","
        st1 = con.createStatement();
     st=con.createStatement();
     String qu1="delete from temp1";
    st.executeUpdate(qu1);
    qu1="select count(*) from clu_t where clu LIKE '%" + to+ "%'";
     rs = st.executeQuery(qu1);
     while(rs.next())
     {
         count=rs.getInt(1);
     }
     if(count==0)
     {
         qu1="insert into temp1 values('"+to+"','0.9')";
                                                st.executeUpdate(qu1);
     }
```

Figure 7.7: Coding snippets for catching the exception

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
HttpSession sn = request.getSession(true);
    String f_name=request.getParameter("name");
   String mail1=request.getParameter("email");
   String pwd=request.getParameter("pass");
   String genter=request.getParameter("gender");
   String cno=request.getParameter("cno");
   String address =request.getParameter("loc");

   String status="null";
```

Figure 7.8: Coding snippets for controlling the server exception

```java
import java.io.IOException;

import com.stanford_nlp.model.SentimentResult;

public class MainApp {

	public static void main(String[] args) throws IOException {

		String text = "Am feel very happy";

		SentimentAnalyzer sentimentAnalyzer = new SentimentAnalyzer();
		sentimentAnalyzer.initialize();
		SentimentResult sentimentResult = sentimentAnalyzer.getSentimentResult(text);

		System.out.println("Sentiment Score: " + sentimentResult.getSentimentScore());
		System.out.println("Sentiment Type: " + sentimentResult.getSentimentType());
		System.out.println("Very positive: " + sentimentResult.getSentimentClass().getVeryPositive()+"%");
		System.out.println("Positive: " + sentimentResult.getSentimentClass().getPositive()+"%");
		System.out.println("Neutral: " + sentimentResult.getSentimentClass().getNeutral()+"%");
		System.out.println("Negative: " + sentimentResult.getSentimentClass().getNegative()+"%");
		System.out.println("Very negative: " + sentimentResult.getSentimentClass().getVeryNegative()+"%");

	}

}
```

Figure 7.9: Coding snippets for storing reviews

```java
		sn.setAttribute("from1",f_name);
	sn.setAttribute("to1",from1);
	 sn.setAttribute("Type1",to1);
		rd=request.getRequestDispatcher("review.jsp");
		rd.forward(request,response);
	 }
	 else if(type1.equals("Map"))
	 {
	     if(to1.equals("Bicycling"))
		{
		 sn.setAttribute("from1",f_name);
	sn.setAttribute("to1",from1);
	 sn.setAttribute("Type1",to1);
		rd=request.getRequestDispatcher("map2.jsp");
		rd.forward(request,response);
		}
		else if(to1.equals("Walking"))
		{
		 sn.setAttribute("from1",f_name);
	sn.setAttribute("to1",from1);
	 sn.setAttribute("Type1",to1);
		rd=request.getRequestDispatcher("map4.jsp");
		rd.forward(request,response);
		}
		else if(to1.equals("Transit"))
		{
		 sn.setAttribute("from1",f_name);
	sn.setAttribute("to1",from1);
	 sn.setAttribute("Type1",to1);
		rd=request.getRequestDispatcher("map3.jsp");
		rd.forward(request,response);
		}
		else if(to1.equals("DRIVING"))
		{
```

Figure 7.10: Coding snippets for storing the mode of transportation

```java
// This class implements a google-like search engine
public class searchEngine {
        // this will contain a set of pairs (String, LinkedList of Strings)
    public HashMap<String,LinkedList<String>> wordIndex;
    public directedGraph internet;             // this is our internet graph

    // Constructor initializes everything to empty data structures
    // It also sets the location of the internet files
    searchEngine() {
        // Below is the directory that contains all the internet files
        htmlParsing.internetFilesLocation = "internetFiles";
        wordIndex = new HashMap<String, LinkedList<String> > ();
        internet = new directedGraph();
    }

    // Returns a String description of a searchEngine
    public String toString () {
        return "wordIndex:\n" + wordIndex + "\ninternet:\n" + internet;
    }

    // This does a graph traversal of the internet, starting at the given url.
    // For each new vertex seen, it updates the wordIndex, the internet graph,
    // and the set of visited vertices.
    void traverseInternet(String url) throws Exception {
        internet.addVertex(url);               //add vertex to map
        internet.setVisited(url,true);  //avoids DDOS by only visiting a webpage

        LinkedList<String> content = htmlParsing.getContent(url);       //parse t
        Iterator<String> itr1 = content.iterator();
```

Figure 7.11: Coding snippets for implementing index value

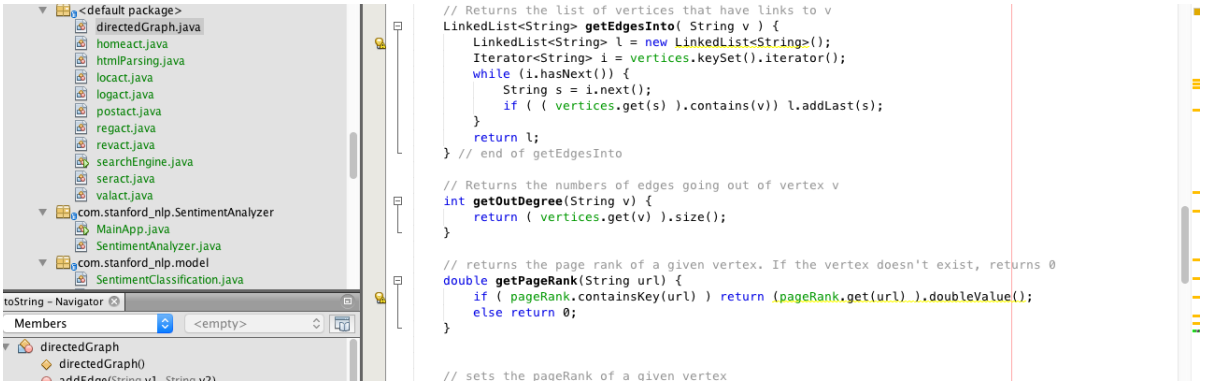

Figure 7.12: Coding snippets for calculating the exact distance commute wise

```
double speed = 15.5; // kmp
double speed_in_meters_per_minute = ( speed * 1000 ) / 60; // mpm

// now calculate time in minutes
 double time1 = kmph / speed_in_meters_per_minute ;
 time=(int)time1;
            }
            else if(t.equals("W"))
            {
           double kmph=distance*1000.0;
double speed = 5.0d; // kmph

double speed_in_meters_per_minute = ( speed * 1000 ) / 60; // mpm

// now calculate time in minutes
 double time1 = kmph / speed_in_meters_per_minute ;
  time=(int)time1;
            }
            else if(t.equals("V"))
            {
           double kmph=distance*1000.0;
double speed = 140.0d; // kmph

double speed_in_meters_per_minute = ( speed * 1000 ) / 60; // mpm

// now calculate time in minutes
 double time1 = kmph / speed_in_meters_per_minute ;
  time=(int)time1;
            }
            else if(t.equals("T"))
            {
           double kmph=distance*1000.0;
double speed = 110.0d; // kmph
```

Figure 7.13: Coding snippets for calculating speed

CHAPTER 8

8.1 Software Modules In Map Android Application

Programming modules being utilized as a part of the Map Android application can be better depicted in type of various exercises being utilized. The conduct or movement is characterized through a class record and a related design. Numerous exercises have been utilized here in order to deal with various utilitarian necessities:

a. Maps Activity.java

Implements the first screen when the app is launched. Takes the origin and the destination i.e clue from the user. Calls all the other modules to find the optimal path for the user to the destination.

b. Application Test.java

Used for the testing purpose of the android application. Contains functions like try and catch for the exceptions which can lead to stopping of the application.

c. Unit Test.java

Implements unit testing of the application. Checks for each function in each and every module and ensures that each function is working properly.

d. Direction Finder.java

Contains the Direction URL API and Google API Key. Finds the most optimal path between the user origin and the clue provided by the user. Communicates with the database and handles the JSON data.

e. Direction Finder Listener.java

Implements the interface Direction Finder Listener. Calls the functions on Direction Finder Start() and on Direction Finder Success().

f. Distance.java

Calculates the distance from the origin and the destination to show on the main page.

Uses the Google API to show the optimal distance.

g. Duration.java

Calculates the duration from the origin and the destination to show on the main page. Uses the Google API to show the optimal duration.

h. Route.java

Chooses the best route from the origin and the destination to show on the main page.Uses the Google API to show the optimal route in many available routes.

i. Activity Maps.xml

This is the layout of the main activity of the application. Contains the elements which define the look of the application. Uses Linear Layout for the Map application.

j. Google Maps API.xml

Contains the Google Map API Key which is used by the Map application. The Key used in this module is taken from the Google Console. The Key is stored as a string datatype in the module which can be easily accessed by the other modules of the application.

CHAPTER 9

CONCLUSION

In this report, we study the problem of clue based route search on road networks, whose main purpose is to find an optimal enough route satisfying all the keywords entered by the user specifically. The matching distance is also taken care of and is cut short. To solve the CRS query, we initiate with greedy clue based algorithm GCS with zero index to narrow it down to the most appropriate candidate to construct an expanded network with an optimal and feasible path.At that point we devise an understood calculation, to be specific piece of information based dynamic programming CDP, to answer the inquiry written by the user that lists out all feasible paths and in return gives out the most optimal result. In order to cut down the computational costs,we propose a branch and bound calculation BAB by applying a specific worldview with the goal that exclusive a restricted measure of the vertices are crossed, in this way enhancing the hunt proficiency. In order to accelerate the speed of finding the search vertices we construct the AB and PB tree. These trees also help in updating a semi dynamic index mechanism. Results from factual studies show that all the proposed algorithm are capable of implementing the CRS query and solving it in an efficient manner. However, the BAB algorithm runs faster and there is a decrease in the size of the index from AB tree to PB tree. There is an extreme scope of research in this field in the coming times. The reason being, users will prefer a more generic POI model as they would like to rate them according to their preference and usage, POI average price of a dress etc, in the query given by the user as a clue.

Secondly, it is good to gather mundane information in order to extend the CRS query.Each purpose of intrigue is doled out with an opening time interval and each intimation has a settled time t that it can use to visit the POI in order to match to the query. The optimised query creates a path so that the the time interval of each matched POI covers the visiting time o the clue. Thirdly, One cannot expect the user to provide the exact keyword a required a they are merely giving the clues which will be inaccurate. Thus, it will be beneficial if the model is able to have an extensive approach towards pro-

cessing of the keywords. It should have a deep search or exhaustive approach.Hence, the matching distance and be changed by integrating both spatial and textual distance together through a linear combinatorial graph.

REFERENCES