Managing road route networks using user-defined keywords

Gopalika Sharma*, Shubham Mehrotra*, Drishti Chauhan*
*B.Tech student, Dept. of Information Technology, SRM University,
Kattankulathur, Chennai, India

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 optimal route covering a sequence of keywords. Although, an optimal route might not always be the road one wants to go through. A personalized query can be defined by some clues that describe the spatial context between Points of Interest along the route 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 propose an AB-tree that stores both the calculated distance and the user entered keyword in the tree structure. In order to diminish the index value we construct a PB-tree by utilizing the hop label index value to land at the pinned location of the user. Extensive and hard-geared experiments are conducted which verify our algorithms.

Keywords - CRS, Spatial keyword queries, clue, Point-of-Interest, travel route search, query processing, CDP, KDD, GCS

I. INTRODUCTION

With the rapid development of location-based services and geo-positioning technologies, there is a clear tendency to make more geo textual objects available in many applications. For example, location information and concise brief descriptions of some companies (eg restaurants, hotels) can easily be found in online local search services (eg, yellow pages). To provide a better user experience, several spatial querying techniques and keywords-related models have emerged, so that geotextiles can be efficiently retrieved. It is common to find a point of interest (PoI) by providing an exact address or a distinguishable key-word (that is, only a few POI contain the specific keyword) in a region that can uniquely identify the position. Some existing jobs extend this query to more sophisticated configurations, such as recovering a group of geo material objects (typically more than 2) or a trajectory that includes multiple keywords.

Unlike your job, our goal is to find a viable solution route in road networks through the use of tracks. In particular, in this document, we have studied a new type of query, ie a search for routes based on clues

CRS, which allows the user to provide clues about text and space context along the path in such a way that a better matching path with the starting point the tracks are returned. More specifically, a CRS query is defined on a G road network, and entry to the query consists of a source vertex vq and a sequence of traces, in which each trace contains a keyword query and a user-provided network distance. A summit contains a keyword key is considered as a vertex of correspondence. The query returns a path P in G that starts in vq, so that passes (i) P through a sequence of corresponding vertices (Poi) w.r.t. the indications and the network distances between two corresponding contagious vertices they are close to the distance specified by the corresponding user in such a way that the user's search intent is satisfied.

A. 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 real scenarios, an optimal route is not required rather a personalized route is preferred. The user may have some requirements while planning a trip. Lets 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 personalized requirements make the route search more distance-sensitive and more flexible such that the distance between the POIs along the route should be as close as specified by the user.

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. Therefore, a novel type of route search which automatically interprets the clues contained in such answers become of vital importance. If we integrate this on our current navigation services, a better user experience can be assured.

B. Problem Statement:

The current answers for trip arranging or course look problem are managing the situations when a client needs to visit a grouping of PoIs, every one of which contains a client indicated watchword. Distinctive enhancement limitations are proposed, and the objective is to locate an ideal course with least cost. In general, the cost can be of different diverse composes, for example, travel separation, time or spending plan. However, to the best of our knowledge, none of the existing solutions on trip planning or route search can be applicable for solving CRS queries since the optimisation needs to be conducted based on the clues. As an extension of traditional route search queries, CRS query can also be useful in many real scenarios.

II. ALGORITHMS BEING USED:

We are going to list out the algorithms which when put together will help us achieve enhanced results. 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.

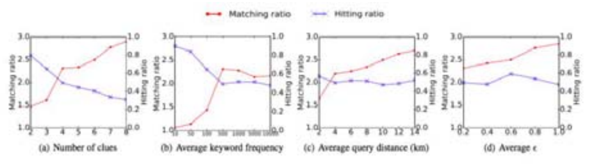


Fig. 5. Accuracy of GCS

Fig. 1. Greedy clue search

A. Clue-based dynamic program algorithm

As we know, 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 an algorithm, called Clue-based Dynamic Programming (CDP). Generally, it is challenging to develop an algorithm for the CRS queries, since the thoroughgoing process to search for POIs in the road networks cannot be avoided .

In CDP, we create a keyword position list for each keyword w(the clue that the user enters), which is a list of vertices that contain w. Whenever a CRS query is generated, the posting lists are sorted according to the keyword . Note that the order of the posting list can be arbitrary as it does not really matter, and hence are sorted by vertex id for simplicity. It is easily noticeable that a k-bipartite graph G has been constructed by these posting lists, which in turn shows all possible paths for a given C. The weight of each edge in G is computed as the matching distance. Specifically, for each u, we define D(wi,u) that denotes the minimum matching distance which can be achieved with a walk that passes the keywords from w1 to wi consistent with the order in C and stops at u. In other words, the weight of vertex u G is computed by D(wi,u), which is the minimum matching distance of all partly feasible paths end at u.

B. 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.

1) 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 generalization while keeping the assumption, since the maximum value of f(x) can be found by the minimum of g(x) = Lf(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) Ld 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.

C. 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 trees 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

III. OVERVIEW OF DATAMINING

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 isnt 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, infor-mation 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 commu-nities 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 prepareing 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.

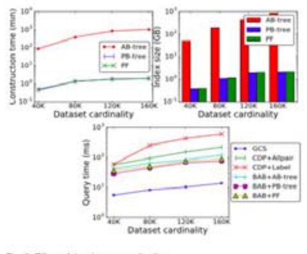


Fig. 8. Effect of the dataset cardinality

Fig. 2. Dataset cardinality

IV. SYSTEM ANALYSIS

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 cal-culations 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:

Users may give heed to a model more suited their needs with POI ratings, like average cost of a hotel room etc. in the query.

It is of interest to take temporal information into account and further extend the CRS query. Each PoI is assigned with an opening hours time interval and each clue has a visiting time t, where the decided query has to find a path so that the interval time matches the interval POI covering the visiting time.

Requiring users to provide exact keyword match is hard sometimes for they are availing a clue which might be not up to the mark.So, it is our imperative to make the model good enough to support the keyword entered by the user approximately.

Advantage:

Useful in job shop scheduling, automatic programming, cir-cuit 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 tech-niques.

V. EXPERIMENTS

A. Experimental Settings

All these algorithms introduced in this report were imple-mented in Netbeans 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 graph that is a part of the road network. The weight of each edge in a graph represents the distance between the two vertices of the edge.

Parameter settings:

In order to make our algorithms run, we generate a random of 100 queries for each set of experiment and thus measure their performance by evaluating their average. 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 demonstrates the percentage of keywords with varied frequencies. In the query, the frequencies of keyword is randomly generated with the average distances and the confidence factors.

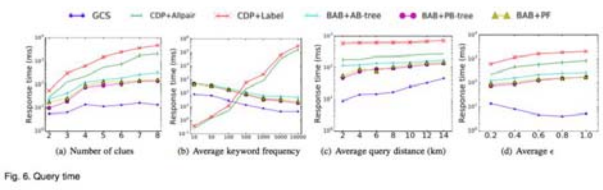


Fig. 3. Query time processing

Performance Evaluation

The performance comparison of all the algorithms are done with the respect of query processing, index size and construction time. The hop label comparison is excluded from our evaluation as of now. In order to venture into query time processing and evaluation, it is easy to find BAB performing better than GCS and CDP. Besides, applying all-pair in CDP has a far shorter response time but cost large costing than utilising hop label, and using PB tree in BAB has a better performance than when one uses AB tree and PF. To take into account, the fact is that for index size and construction time, label based approaches have fairly much smaller size and less time than all pair based approaches.

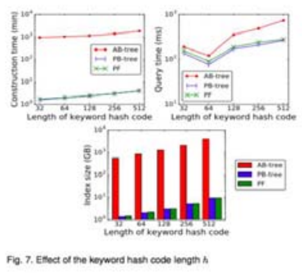


Fig. 4. Hash effect

VI. PRACTICAL IMPLEMENTATIONS

A. 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.

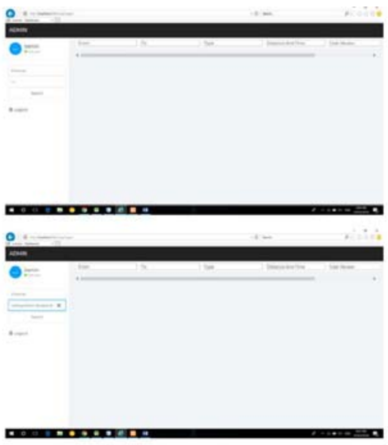


Fig. 5. Entering keywords

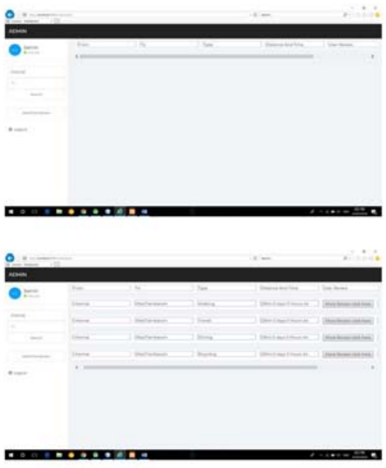


Fig. 6. Modes of Transportation

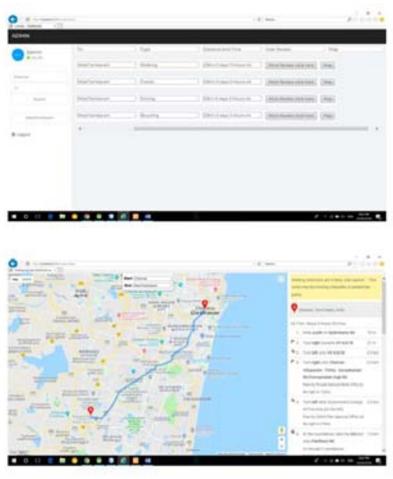


Fig. 7. Map with directions

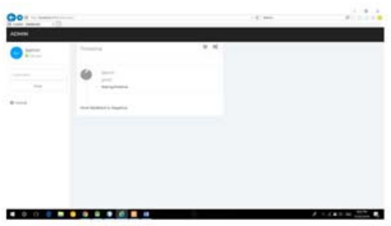
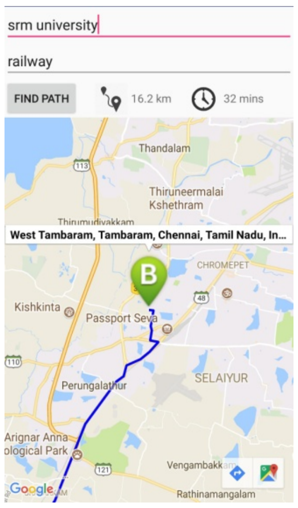


Fig. 8. Review page

B. 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.



srm university

railway

FIND PATH 16.2 km 32 mins

West Tambaram, Tambaram, Chennai, Tamil Nadu, In...

Fig. 9. Running of the app

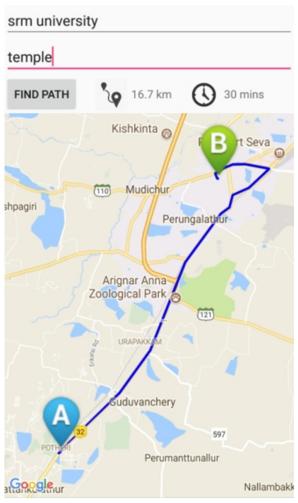


Fig. 10. Implementing of keywords

VII.RELATED WORK

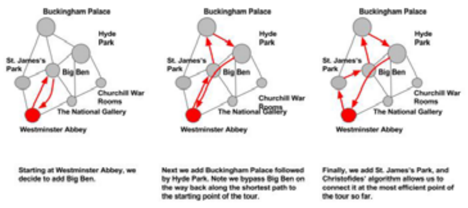
A. Travel Route Search

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.

Then comes into being the LORD or R-LORD algorithms to filter out the locations which do not deem fit as locations in the optimal route, thus this improves our search efficiency. The problem in studies of multi-rule partial sequence route (MRPSR) has been identified. This aims to find an optimal route with minimum distance under a rule which supports partial category order defined in the query. They propose three heuristic algorithms to search for utmost optimal solutions for the user given query. This henceforth gives birth to the idea of a greedy algorithm which finds a route whose length is smaller than a specified threshold whereas the text relevancy is up to its maximum.

The problem of finding a path which touches at least one satisfying entity of each type is definitely an interactive approach.In each step, a user is supposed to give its feedback as to whether he was satisfied by the entity or not. Hence, the problem of multi-keyword routing query in an approximate manner takes place which complements the standard way of finding the shortest path accompanied by multiple keywords and a string similarity function. For each keyword, the match-ing point is supposed to have an altered distance smaller than the mentioned threshold. It defines the problem of keyword aware optimal route query, which is to find a route which covers each and every user specified keyword, a specific budget and the objective score should be taken into consideration. It proposes two different solutions, namely backward and forward searches, in order to deal with the general and the most optimal route query with a total order. It brings in light the problem of personalised trip recommendation, which aims to find the most favourable trip that accelerates the user experience for a stipulated time and a budget constraint and also takes the uncertainty of travelling time into play.



Starting at Westminster Abbey, we decide to add Big Ben.

Next we add Buckingham Palace followed by Hyde Park. Note we bypass Big Ben on the way back along the shortest path to the starting point of the tour.

Finally, we add St. James's Park, and Christofides' algorithm allows us to connect it at the most efficient point of the tour so far.

Fig. 11. Example for Travel route search

VIII. CONCLUSION AND FUTURE ENHANCEMENTS

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.Then we devise an implicit algorithm, namely clue-based dynamic programming CDP, to answer the query 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 algorithm BAB by applying a particular paradigm so that only a limited amount of the vertices are traversed, thus improving the search efficiency. 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 much 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 point of interest is assigned with an opening time interval[T] and each clue has a fixed 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 processing of the keywords. It should have an 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

[1] I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck. Hierarchical hub labelings for shortest paths.In ESA, pages 2435. Springer, 2012.
[2] T. Akiba, Y. Iwata, K.-i. Kawarabayashi, and Y. Kawata. Fast shortest path distance queries on road networks by pruned highway labeling.In ALENEX, pages 147154. SIAM, 2014.
[3] J. L. Bentley and J. B. Saxe. Decomposable searching problems i. staticto-dynamic transformation.Journal of Algorithms, 1(4):301358, 1980.
[4] X. Cao, G. Cong, C. S. Jensen, and B. C. Ooi. Collective spatial keyword querying.In SIGMOD, pages 373384. ACM, 2011.