ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 5, Issue 7, July 2016

Efficient method for Online Shortest Path Computation on Time Dependent Networks

Mr. Nikhil Asolkar
Computer Engineering Department
JSPM's Imperial College of Engineering And Research
Pune,India

Prof. S.R.Todmal
Computer Engineering Department
JSPM's Imperial College of Engineering And Research
Pune,India

***Abstract*— For computing the shortest paths, earlier methods used offline data which were pre-stored in navigation systems and travel time as weight of the road edges to estimate road distance. However the circumstances of road traffic are changing over the time. Hence the results of shortest path computation system are not considered as accurate result for longer time without considering the live traffic data. To overcome this problem of online shortest path computations many recent methods of computing the shortest paths were based on live traffic data. These methods had limitations of extract costs required for continuous online data monitoring from spatial databases and scalability etc. Here we are proposing efficient method called extended live traffic index which allows drivers to collect the live traffic data quickly and effectively over the broadcasting channel. ELTI is considering the time-dependent networks, in which the travel time along each arc is an unknown function of the departure time.**

***Index Terms*—Shortest path, air index, broadcasting**

I. INTRODUCTION

Shortest path computation is an imperative function in current car navigation systems to discover the best route for a driver from his existing position to destination. Characteristically the undeviating path is calculated by offline information pre-stored within the navigation systems and consequently the weight (travel time) of the road edges is calculable by the road distance or historical information. Regrettably, road traffic circumstances modify in due course of time. Whilst the live traffic circumstances do not, thus, the route that is projected by the navigation system is unreliable and insecure.

Lately, many online services have been offering live traffic information like GoogleMap, Navteq, INRIX Traffic info supplier, and TomTom NV, etc. These systems will estimate and calculate the shortest path queries supported by the current live traffic data; but, as a consequence of high operational costs they are doing not report routes to drivers unceasingly. Respondent the shortest paths on the live traffic information are frequently observed as an eternal observance downside in spatial databases explicitly termed online shortest paths computation (OSP) during this work. To the most effective of our information, this shortcoming has not received a lot of consideration and also the costs of respondent, such continuous queries differ particularly in numerous system architectures.

Characteristic client-server design will be accustomed answer shortest path queries on live traffic information. All throughout this case, the navigation system habitually sends the shortest path query to the service supplier and waits for the result back from the supplier. Nevertheless, given the ascension of mobile devices and services, in terms of network bandwidth and server loading, this model is facing measurability inadequacies.

A innovative and promising resolution to the shortest path computation to broadcast an air index over the wireless network is Live traffic index (LTI) which is expected to offer relatively short tune-in cost (at client side), fast query response time (at client side), small broadcast size (at server side), as well as light maintenance time (at server side) for OSP.

Two novel techniques optimize the index structure of LTI that is graph partitioning and stochastic-based construction, when performing an intensive study on the hierarchic index techniques. To the most efficient of our information, this can be the key work to present an intensive cost examination on the hierarchic index techniques and apply framework to optimize the index hierarchical data structure.

By integrating Dynamic Shortest Path Tree (DSPT) into hierarchic index techniques, LTI with effectiveness maintains the index for live traffic circumstances. Moreover, a predetermined version of DSPT is projected to more scale back the published overhead.

LTI diminishes the tune-in cost up to an order of magnitude as compared to the state-of-the-art competitors by integrating the above aspects; whilst it still provides competitive query response time, broadcast size, and maintenance time.

For computing the shortest ways, earlier strategies uses the offline information that is pre stored in navigation strategies additionally as period as weight of the road edges to estimate the historical information or road distance. But the circumstances of road traffic are dynamical over the time. Thus the results of shortest path computation system are not thought of as correct result for extended time while not considering the live traffic information. To beat this problem of on-line shortest path computations is later made-up but these methods are having limitations of extract prices needed for continuous on-line information observation from special databases and measurability etc. during this project we tend to proposing economical technique new framework referred to as extended live traffic index (ELTI) that permits drivers to gather the live traffic information quickly and effectively over the broadcasting channel. This approach is based on recent technique referred to as LTI. ELTI is considering the time-dependent networks, within which the period on each arc is unknown perform of the time of departure on the arc for economical computation of shortest path.

In algorithmic graph theory, the static shortest path problem is one amongst the foremost studied issues. In reality however, for computing shortest ways, several networks tend to own dynamic characteristics that need a lot of sophisticated approaches. For dynamic shortest path issues, there are two common types: because of frequent, fast, and unpredictable changes in network information, one should recompute shortest ways. For finding a series of closely-related static shortest path issues, this can be basically the optimization problem. The subsequent one, and therefore the study of this paper, is that the time-dependent shortest path problem, throughout which network characteristics modify with time in a very assured manner in transport transportation, such concerns ascend from time to time; in arc travel times that are definite to occur joined travels through the network, if one considers certain future changes, significantly around times like “rush hour", the shortest path calculated from a photograph of the network information at the current time might not be optimum. Within the time-dependent shortest path problem, we have a tendency to assume that the period of time on every arc may be operate of the time of departure on the arc, and ahead over all time, that each one such functions are identified. In distinct time, the matter at first dates back to 1966, once it had been rest projected.

To solve a range of NP-Hard improvement issues like the knapsack problem, the final time-dependent shortest path drawback is a minimum of NP-Hard since it's going to be used. Nevertheless, betting on however one denies the matter, since its output isn't polynomial delimited, it's going to not be in NP; furthermore, there are even continuous-time instances during which shortest ways carries with it an in note sequence of arcs. During this paper, we have a tendency to study a special category of networks called FIFO networks, during which a First-In-First-Out manner, commodities travel arcs. In apply several networks, significantly transportation networks, exhibit FIFO behavior. Under the FIFO assumption, for economical polynomial-time resolution algorithms time-dependent shortest path issues, exhibit several nice structural properties that alter the event. Our goal during this discussion is to review the theoretical properties and resolution algorithms of time-dependent shortest path issues at intervals. A straightforward framework that unites previous ends up in each continuous and distinct time. At intervals, this framework, for problem variants we have a tendency to contemplate a good vary.

To describing the time-dependent shortest path problem and its variants, the remainder 2 sections are devoted. By associate degree abundance of problem variants, creating a concise problem description rather difficult, time-dependent shortest path issues are troubled. For these 2 variants, we have a tendency to show a way to reduce the house of problem variants all the way down to 2 basic variants, and that we proceed to present properties and resolution algorithms. Finally, for the non-FIFO case at intervals the context of the ends up in this paper, we have a tendency to concisely discuss results.

II. Literature Survey

In this section we are presented the review of different methods presented for mining high utility item sets from the transactional datasets.

In [1], two executions of this idea were presented by H. Bast, S. Funke, D. Matijevic, P. Sanders, and D. Schultesone on a modest griddata structure and one centered on highway hierarchies. For the road map of the United States, our best query time advance over the best previously published figures by two orders of magnitude. The results display several trade-offs amongst average query time (5µsto 63µs), preprocessing time (59 min to 1200 min), and storage overhead (21 bytes/node to 244 bytes/node).

In [2], a shortest path algorithm that permits fast point-to-point queries in graphs by means of preprocessed data was presented by P. Sanders and D. Schultes. Here, we give abroad revision of our technique. It consents quicker query and pre-processing time, it diminishes the size of the data gained during the preprocessing and it deals with directed graphs. Some imperative conceptions like the neighborhoods radii and the retrenchment of a network have been generalized and are now further flexible. The query algorithm has beenabridged: from the bidirectional version of Dijkstra’s algorithm, it contrasts only by a few lines. Even if the graph covers several paths of the same length, we can show that our algorithm is accurate.

In [5], B. Jiang familiarizes the representation and a model of a paging environment. The I/O efficiencies of the single-source selected, all pairs, and multi-source algorithms are deliberated and scrutinized.

In [7], a vibrant technique for fast route planning in large road networks is presented by P. Sanders and D. Schultes. For the first time, it is probable to handle the practically relevant situations that ascend in present-day navigation systems: After an edge weight changes (e.g., owing to a traffic jam), we can apprise the preprocessed information in 2–40 ms permitting successive fast queries in around one millisecond on an average. It skips a reasonably expensive apprise step and directly execute a prudent query that mechanically takes the altered state into account, when we intend to achieve only a single query. Also pre-computing the preprocessed information takes characteristically less than two minutes if the whole cost function changes (e.g., because of the different vehicle types).

In [8] an unusual query processing technique was introduced by W.-S. Ku, R. Zimmermann, and H. Wang, which by preserving high scalability and accuracy, managed to diminish the latency considerably in answering LBSQs. This method is grounded on peer-to-peer sharing, which enables us to process queries by means of query results cached in its neighboring mobile peers without delay at mobile host. It exhibits the practicability of method through a probabilistic analysis, and exemplifies the appeal of the method through extensive simulation results. Index Terms—Broadcast disks, mobile computing, mobile environments, location dependent and sensitive.

In [9] to considerably accelerate processing of the set of designated routes specified by incessant route planning queries in the face of incoming traffic delay updates N. Malviya, S. Madden, and A. Bhattacharya propositioned two novel classes of approximate methods, K-paths and proximity measures. This methods work through an amalgamation of pre-computation of probable good paths and by avoiding absolute recalculations on each delay update, as an alternative only sending the client new routes after delays change notably. Centered on an experimental evaluation with 7,000 drives from real taxi cabs, it established that the routes conveyed by our techniques are within 5% of the best shortest path and have run times an order of magnitude or less contrasted to an inexperienced approach.

In [10], wireless mobile environments Y. Jing, C. Chen, W. Sun, B. Zheng, L. Liu, and C. Tu represents wireless broadcast which conveys a scalable and safe spatial data dissemination approach for geographical applications. Amid the numerous location-based services, the shortest path query on road networks is one of the most widespread and indispensible services in our everyday life. Through this study and paper, we recommend an energy-efficient scheme for on air shortest path query processing on road networks, which influences an elaborate air index called Bag Index centered upon the innovative Hilbert-based heuristic tree decomposition for the road networks. Investigational results demonstrate that the projected approach sustains a reduced amount of energy consumption on both communication and computation than the fore mentioned schemes.

III. Framework and Design

3.1 Architecture

In current car navigation methods shortest path computation playing important role and has been extensively studies in recent many researches. Such functions of computing shortest paths are giving the optimal route from current location to destination position by computing the shortest path either by offline traffic dataset or live traffic dataset to select best route. The current researchers are mainly focusing on live traffic data for computing the shortest paths using online services like Google Map, Navteq etc. but these approaches writhe from the limitations of not continuously providing routes to the drivers due to higher operational costs. This problem is commonly known as continuous monitoring problem in spatial databases. To overcome this problem, recently many methods were proposed, but still they suffered from the various limitations like scalability, more tune-in costs, query response time is more, more maintenance time etc. The recent approach studied called live traffic index (LTI) overcomes all these previous methods and their problems, but this approach is not implemented over time dependent networks.

Below figure 1 shows the proposed system block diagram and details of algorithm proposed.

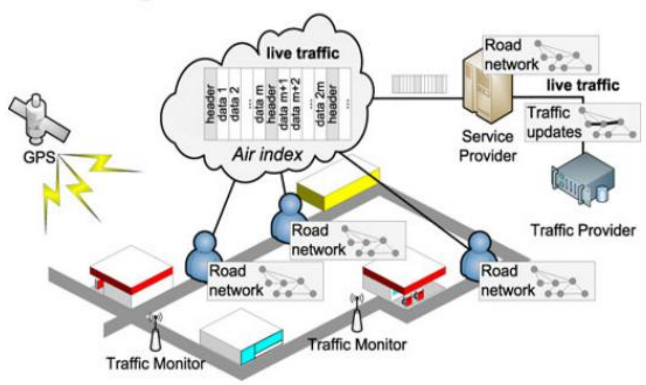

Figure 1: System Architecture

3.2 Process Flow:

Our approach in this paper displays a system architectural overview from the traffic monitors by means of techniques like road sensors and traffic video exploration. The live traffic circumstances collects by the traffic provider from the traffic provider and broadcasts, the live traffic index on radio or wireless network (e.g., 3G, 4G, Mobile WiMAX, LTE, etc.) receives by the service provider periodically. It attends to the live traffic index and recites the relevant portion of the index for calculating the shortest path to and monitor a shortest path wishes by a mobile client on handling traffic updates, we focus but not graph structure updates. For genuine road networks, when equated to edge weight updates (i.e. live traffic circumstances), to have graph structure updates (i.e., construction of a new road) it is sporadic. Thus through typical transmission protocol (i.e., HTTP and FTP) in advance (e.g., by monthly updates or at system boot-up) we accept that the graph structures are circulated to all clients.

3.3 Mathematical Model:

Step 1:

Cost analysis: The total space requirement of a hierarchical indexIcan be represented as follows:

$$|I| = \sum_{SG_i \in I} (|V_{SG_i}| + |E_{SG_i}| + |T_{SG_i}| + |\Delta_{SG_i}|) + tree,$$

Where V_{SG_i} and E_{SG_i} Represents the nodes and edges in SG_i,Resp, T_{SG_i}represents the connectivity information between the child entries, Δ_{SG_i} represents the pre-computed information kept in SGi, and tree represents the hierarchical information of I. Since VSGi , ESGi and GSGi are directly derived from the original graph andtreeis negligible compared toG, the space requirement can be revised as the follows:

$$|I| = |G| + \sum_{SG_i \in I} |\Delta_{SG_i}|.$$

Step 2:

To minimize the index broadcast size, it is more or less equivalent to minimize the size of ΔSGi . The simplest way is to partition the graph into multiple subgraphs such that the total size of ΔSGi is minimized. However, this may not optimize the query performance being discussed shortly.

Our objective is to find a hierarchical index structure I such that

$$OBJ(I) = \min_{SG_i \epsilon I} \frac{\Sigma|\Delta_{SG_i}|}{\min\{|V_{SG_i}|\}},$$

Where min{VSGi} can be viewed as a normalized factor such that the objective function prefers balanced partitions.

$$OBJ_{Cheegr}(G) = \min_{SG_1 \ldots\ldots SG_n \epsilon G} \frac{Cut(\{SG_1 \ldots., SG_n\})}{\min\{|SG_1|, \ldots \ldots .|SG_n|\}},$$

Where $Cut(\{SG_1 \ldots., SG_n\})$ is the number of edges between any two subgraphs.

Step 3:
NP-Hard and NP-Complete Analysis:-

NP-hard means, to different problems considering Problem A which can be solved many times by reducing it. If given a solution to Problem A, I helps in reducing Problem B to Problem an, to Problem B, I can easily construct A solution.

I can reduce any problem in NP to that problem, if a problem is NP-hard. I can easily solve the problem in NP, this means if I can solve that problem, this would prove P = NP, if we could solve an NP-hard problem in Polynomial time.

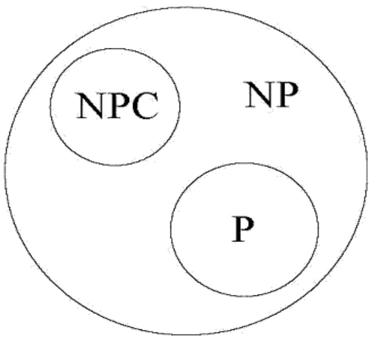

Figure 2: NP hard

Technically, O (n) actually means the algorithm runs in asymptotically linear time, which means the time complexity approaches of a line as n gets very large. Also, O (n) is theoretically an upper bound, so if the algorithm runs in sub linear time we can say it's O (n), even if its not the best description of the problem.

Step 4:
Deterministic finite automaton:

A deterministic finite automaton M is a 5-tuple, (Q,∑, δ,q0, F) consisting of

- A finite set of states (Q) = {0, 1, 2, 3, 4}
- a finite set of input symbols called the alphabet(∑)={esp:esp, a:esp, d:c, b:b}
- a transition function (δ: Q X∑ → Q) = { }
- a start state (q0 εQ)={q0}
- a set of accept states (F ⊆ Q) = {q1}

where
0= initial State
1= final state
EPS=Elementary Shortest Path

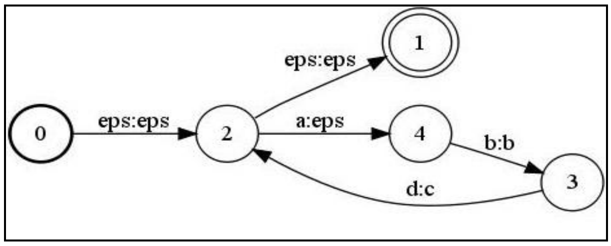

Figure 3: State transition diagram

Step 5:
Cost-matrix and flag multiplexers (MUX unit):

The MUX unit is the chief unit of the processing block of the shortest path processor is. It comprises two 2-to-1multiplexers work in parallel. The first one is accustomed to pass either the initial cost matrix provided from latch block or the resulted cost matrix provided from the intermediate unit of the processing block. The second multiplexer is used in the similar way however, for passing either the initial flag vector or the updated flag vector approaching from the final processing unit. Two supplementary inputs supplied to the two multiplexers, selector signal for selecting one of the two multiplexer inputs and the enable signal for enabling the two multiplexers. The selector is created by OR gate, all bits of the updated flag vector which is initiated to zero value. So, the first repetition of the processing block will supplied by the initial cost matrix and flag vector. After the first repetition is accomplished, at least two of the updated flag vector bits will be transformed to 1 and the output of the OR gate will also be transformed to 1. Consequently, the second and the succeeding repetitions will be supplied by the updated cost matrix and flag vector. The enable signaling driven from the latter latch signal with one clock delay is to assure that the initial cost matrix as well as flag vector extent to the multiplexer's inputs.

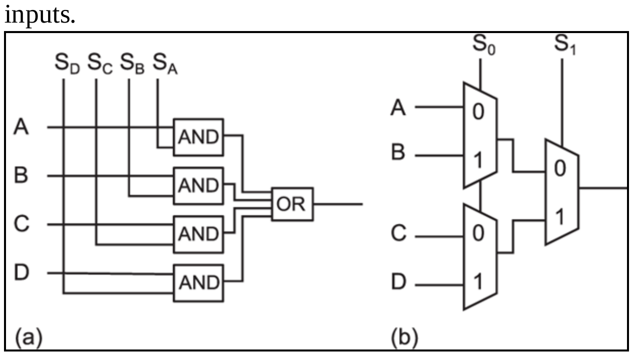

Figure 4: Multiplexer Logic

IV. WORK DONE

In this section we are discussing the input dataset, system requirement, practical environment, scenarios, performance metrics used etc.

4.1 Input:

In this datasets, source node and destination node are the input for our practical experiment.

4.2 Hardware Requirements:

- Processor: Intel core i3, i5
- RAM: 1GB
- Disk: 20 GB
- Monitor : Any Color Display
- Standard Keyboard and Mouse

4.3 Software Requirements:

- Operating System : Windows 7
- JDK: 1.7.0
- Database Server: My Sql
- NetBeans IDE: 7.0

4.4 Results of Practical Work:

Following figures are showing results for practical work which is done. Following figures shows the main screen. That takes the input data set,

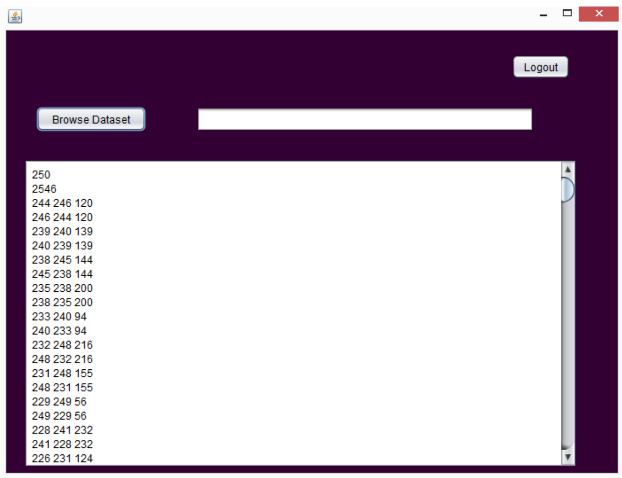

Fig.5: Browse the dataset.

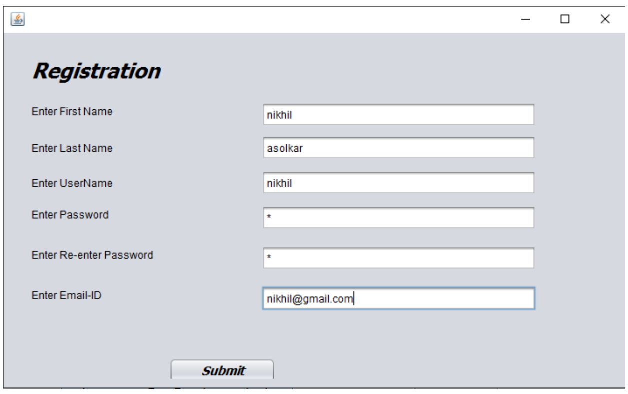

Fig. 6: Registration of New User

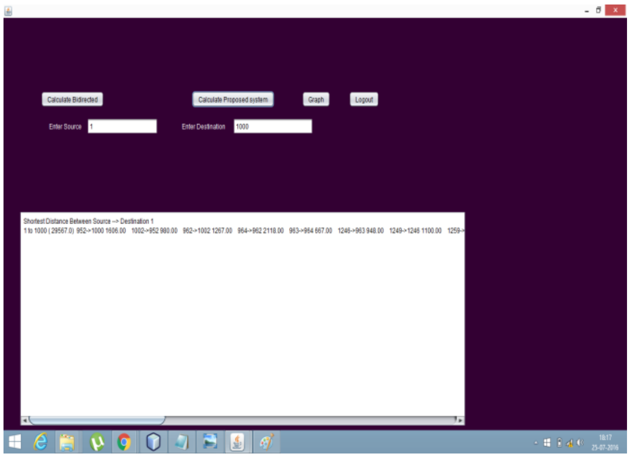

Fig 7: Output: Source to destination

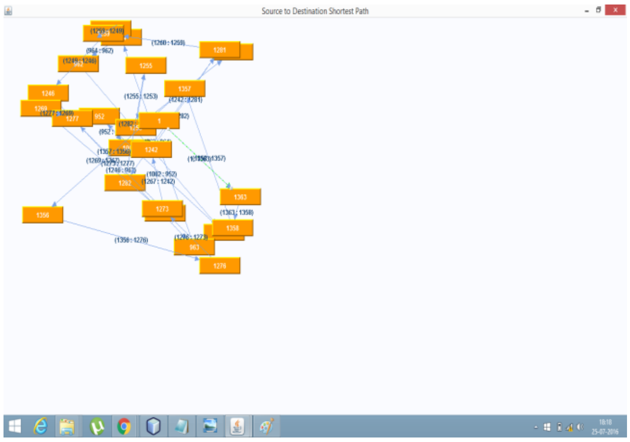

Fig 8: Output: Source to destination nodes

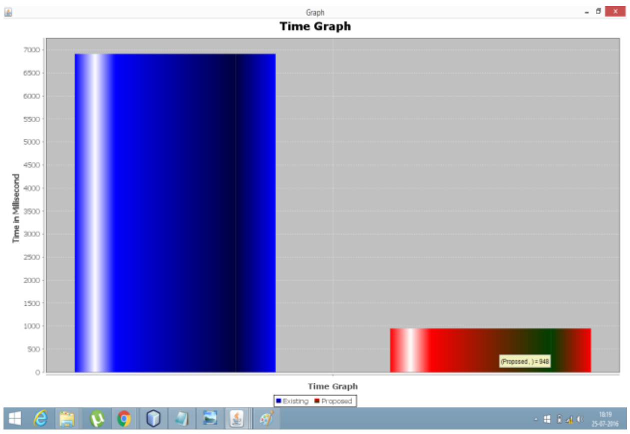

Fig 9: Time graph

4.5 Comparing the proposed & existing system:

The comparison of existing and proposed system is shown in the graph below. The graph is constructed by numerical values considering the time required to solve the query for calculating the shortest path from source to destination. The graphical result clearly shows that proposed system is superior to existing system.

ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 5, Issue 7, July 2016

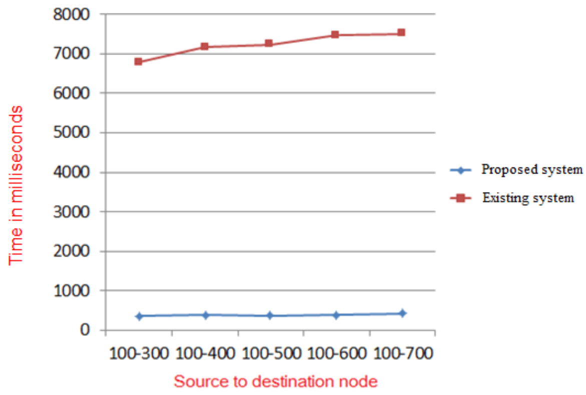

Fig 10: Timeline: proposed & existing system

V. Conclusion and Future Work

In this paper we have studied online shortest path computation; centered on live traffic situation which aids the shortest path result's computed/updated. We have rigorously analyzed the present work and discussed their peripheral nature to the problem (owing to their preventive maintenance time and enormous transmission overhead). To deal with the setback, we recommend a capable design that broadcasts the index on the air to figure out shortest path on a small portion of index. We tend to initially establish a vital feature of the stratified index structure that permits us in our resolution, LTI. This necessary feature is totally used for online shortest path computation. Our experiments make sure that LTI may be a pareto optimal solution in terms of four performance factors. In the proposed system, we are going to extend our resolution on time dependent networks, which depends not only on current traffic information however conjointly supports the expected traffic circumstances for computing shortest path.

References

[1] H. Bast, S. Funke, D. Matijevic, P. Sanders, and D. Schultes, "In Transit to Constant Time Shortest-Path Queries in Road Networks," Proc. Workshop Algorithm Eng. and Experiments (ALENEX), 2007.

[2] P. Sanders and D. Schultes, "Engineering Highway Hierarchies," Proc. 14th Conf. Ann. European Symp. (ESA), pp. 804-816, 2006.

[3] G. Dantzig, Linear Programming and Extensions,series Rand Corporation Research Study Princeton Univ. Press, 1963.

[4] R.J. Gutman, "Reach-Based Routing: A New Approach to Shortest Path Algorithms Optimized for Road Networks," Proc. Sixth Workshop Algorithm Eng. and Experiments and the First Workshop Analytic Algorithmics and Combinatorics (ALENEX/ANALC), pp. 100-111, 2004.

[5] B. Jiang, "I/O-Efficiency of Shortest Path Algorithms: An Analysis,"Proc. Eight Int'l Conf. Data Eng. (ICDE), pp. 12-19, 1992.

[6] P. Sanders and D. Schultes, "Highway Hierarchies Hasten Exact Shortest Path Queries,"Proc. 13th Ann. European Conf. Algorithms (ESA), pp. 568-579, 2005.

[7] D. Schultes and P. Sanders, "Dynamic Highway-Node Routing," Proc. Sixth Int'l Conf. Experimental Algorithms (WEA), pp. 66-79, 2007.

[8] W.-S. Ku, R. Zimmermann, and H. Wang, "Location-Based Spatial Query Processing in Wireless Broadcast Environments,"IEEE Trans. Mobile Computing, vol. 7, no. 6, pp. 778-791, June 2008.

[9] N. Malviya, S. Madden, and A. Bhattacharya, "A Continuous Query System for Dynamic Route Planning,"Proc. IEEE 27th Int'l Conf Data Eng. (ICDE), pp. 792-803, 2011.

[10] Y. Jing, C. Chen, W. Sun, B. Zheng, L. Liu, and C. Tu, EnergyEfficient Shortest Path Query Processing on Air,"Proc. 19th ACM SIGSPATIAL Int'l Conf. Advances in Geographic Information Systems (GIS), pp. 393-396,2011. "Google Maps," http://maps.google.com, 2014.

[11] "NAVTEQ Maps and Traffic," http://www.navteq.com, 2014.

[12] "TomTom NV," http://www.tomtom.com, 2014.

Mr. NIKHIL ASOLKAR
PG Scholar, Department of Computer Engineering, JSPMs Imperial College of Engineering and Research, Pune, India.

Prof. S.R.TODMAL
HOD, Department of Computer Engineering, JSPMs Imperial College of Engineering and Research, Pune, India.