International Journal of Advanced Technology and Innovative Research

www.ijatir.org

ISSN 2348–2370
Vol.07,Issue.18,
December-2015,
Pages:3482-3487

Cellular Automata Index Based Construction for Shortest Path Computation

M. HEMA BHAVANI[1], DR. P. KIRAN SREE[2]

[1]PG Scholar, Dept of CSE, Sri Vishnu Engineering College for Women, Bhimavaram, AP, India,
E-mail: hemavsm26@gmail.com.

[2]Professor, Dept of CSE, Sri Vishnu Engineering College for Women, Bhimavaram, AP, India,
E-mail: pkiransree@gmail.com.

Abstract: The shortest direction problem aim at computing the shortest route primarily based on traffic occasions. That is very crucial in contemporary automobile navigation systems because it enables drivers to make sensible decisions. To our best knowledge, there may be no efficient machine/answer which could provide low-priced costs at both customer and server aspects for shortest route computation. The traditional customer-server structure scales poorly with the range of customers. A promising approach is to let the server gather live site visitors records after which broadcast them over radio or wireless community. This approach has exquisite scalability with the range of customers. A hierarchical index shape CAI that achieves all optimization goals is built with cellular Automata. For that reason, we broaden a brand new framework referred to as Cellular automata traffic index(CATI) which enables drivers to fast and efficiently collect the live site visitors records at the broadcasting channel. An impressive end result is that the motive force can compute/update their shortest route result through receiving only a small fraction of the index. Our experimental study suggests that CATI is robust to various parameters and it offers especially short song-in fee (at consumer aspect), rapid question reaction time (at consumer aspect), small broadcast size (at server aspect), and light renovation time (at server aspect) for shortest direction trouble.

Keywords: TI-TD, Shortest Path, Transmission Overhead.

I. INTRODUCTION

Shortest path computation is a very important mechanism in modern automotive navigation systems. This mechanism helps a driver to get the simplest route from his current position to destination. Typically, the shortest path is computed by offline information which was already stored within the navigation systems and also the waiting (travel time) of the road edges is calculate by the road distance or already stored information. Sadly, road traffic circumstances modification over time. While not live traffic circumstances, the route came back by the navigation system Associate in isn't any longer secured and correct result. Computing shortest ways could be a most preferable operation for any issues in traffic applications. The most outstanding area unit definitely route coming up with systems for cars , bikes and hikers , or timetable info systems for scheduled vehicles like trains and buses. If such a system is given as a central serve r, it's to ans we tend to a huge number of client queries soliciting for their best itineraries. Users of such a system ceaselessly enter their requests for locating their “best” connections. Furthermore, similar queries seem as sub-problems in line coming up with, time table generation, logistics, and traffic simulations. Nowadays, many on-line services give live traffic information such as Google-Map, Navteq , INRIX Traffic info Provider.

On-line services analysed collected information from road sensors, traffic cameras. These systems area unit able to figure shortest path supported current live traffic information. However they are doing not give routes to drivers. Traffic information provides info concerning speeds on specific road ever-changing over time. It's vital in network analysis. Traffic affects travel times, that successively have an effect on results hence network analysis is vital. If you're coming up with a route from one place to a different and while not considering traffic, expected travel and arrival times couldn't be correct. You will miss routes that save time by avoiding the slower and full roads. With the recognition of on-line map applications and their wide deployment in mobile devices and car-navigation systems, an increasing variety of users search for point-to-point most quick paths and also the corresponding travel-times. This drawback has been extensively studied on static road networks wherever edge costs area unit constant. Several economical speed-up techniques have been developed to figure the quickest path in a very matter of milliseconds. The fastest path approaches create the assumption that the travel-time for every range of the road network is constant. In real-world the particular travel-time on a road heavily depends on the holdup and, therefore, it is time-dependent.

One will observe that the time-dependent travel-times yield a modification within the actual fastest path between any try of places throughout the day. Specifically, the quickest path from one place to a different varies reckoning on the departure-time from the supply. Cellular automata (CA) is dynamical systems which are discrete in space and time, operate on a uniform, regular lattice and characterised by "local" interactions. Cellular automata provide a formal framework for investigating the behavior of complex,

extended systems.CA are dynamical systems in which space and time are discrete. A cellular automata consists of a regular grid of cells, each of which can be in one of a finite number of possible states, updated synchronously in discrete time steps according to a local, identical interaction rule. The state of a cell is determined by the previous states of a surrounding neighborhood of cells.

II. RELATED WORK

In the last decade, various economical shortest path algorithms with pre-computation strategies are planned for computing the shortest paths. However, the area unit some studies that focus on economical computation of time-dependent shortest path drawback. Cooke and Halsey 1st studied the time-dependent shortest path (TDSP) drawback wherever they resolved the matter by formulating it in discrete-time and exploitation Dynamic Programming. Another discrete-time resolution to TDSP is to utilize time-expanded networks Ref[1]. In general, time-expanded network approaches assume that the sting weight functions area unit outlined over a finite distinct window of your time t ∈ t0, t1, .., tn, wherever American state is decided by the total period of your time interval into consideration. Therefore, the problem is reduced to the matter of computing, for every time window, minimum-weight ways through the static network wherever one will apply any of the well-known shortest path algorithms. Although these algorithms area unit simple to style and implement, they have various shortcomings Ref[2]. First, time-expanded models want to create a separate instance of network for every time instance hence leading substantial quantity of storage. Second, such approaches can them selves offer approximate results as a result of the model misses the state of the network between any 2 discrete-time instants. Finally, it's terribly exhausting to come to a decision on the effective selection of discrete-time intervals for real-world applications. George and Shekhar planned time-aggregated graph approach wherever the time-dependent travel-times of every edge area unit aggregative into time series. Albeit this model needs less house than that of time expanded networks, the results area unit still approximate Ref[3].

Army officer showed that the time-dependent shortest drawback can be resolved by a generalization of Dijkstra's technique as efficiently as for static shortest path issues. Yet, Halpern proved that the generalization of Dijkstra's formula is barely true for inventory accounting networks. If the inventory accounting property doesn't hold in an exceptionally time dependent network, then the stuff is NP-Hard Ref[4]. Orda and Rom introduced a time-dependent shortest path approach supported Bellman-Ford formula. With their approach, they verify the path toward destination by refinement the arrival-time functions on each node with in the whole quantity T. In Kanoulas et al Ref[5]. Introduced Time-Interval All quickest Path (allFP) approach within which they maintain a priority queue of all ways to be dilated instead of sorting the priority queue by scalar values. Therefore, they enumerate all the ways from the supply to a destination node that incurs exponential time period within the worst case Ref[6]. The elevation formula was planned to accelerate shortest path computation in static road networks. With ALT, a collection of nodes known as landmarks area unit chosen so the shortest distances between all the nodes within the network and every one the landmarks area unit computed and stored. elevation employs triangle difference supported distances to the landmarks to get heuristic operate to be utilized in A* search. The time-dependent variant of elevation is studied in wherever heuristic function is computed w.r.t lower-bound graph. The Contraction Hierarchies (CH) and SHARC strategies (also developed for static networks) were increased to time dependent road networks Ref[7].

III. PROPOSED METHOD

A. Towards Time-Dependent Path Planning

Cellular automata is a regular lattice (repeated structure of points have the same kind of neighborhood) one puts a finite-state machine at each point. The input to the machine is the states of all machines in its neighborhood. The behaviour is to change its state based in a determined way, as a function of the states of its neighbors and its own state. The states of all machines in the lattice are updated synchronously (simultaneously). The travel-times that take into consideration the traffic conditions ar merely computed by considering increased edge weights (that corresponds to traffic congestion) for every path. However, our time-dependent path designing leads to completely different optimum methods for different departure-times from the supply. for instance, contemplate wherever Google Maps supply two various methods (and their travel-times underneath no-traffic and traffic conditions) for associate origin and destination try in la road network.

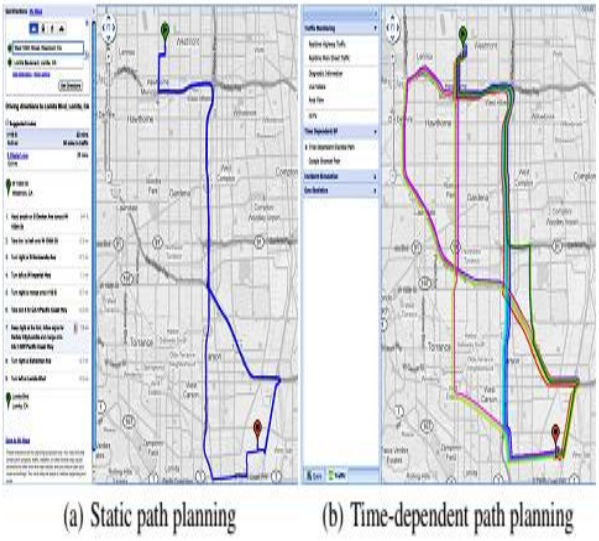

Fig.1.(a) Static path planning, (b)Time-dependent path planning.

Note that the trail recommendation and therefore the travel-times stay identical despite once the user submits the

question. Fig 1a static path planning On the opposite hand, Fig1b time-dependent path planning recommendations (in completely different colours for various departure times) for identical origin and destination try wherever we have a tendency to computed the time-dependent quickest methods for thirty eight consecutive departure-times between 8:00 AM and 5:30 PM, spaced quarter-hour apart. As shown, the best methods amendment throughout the course of the day. One could argue adjacent to the opportunity of time-dependent path designing algorithms due to a) inconvenience of the time-dependent edge travel-times, or b) negligible gain of time-dependent path designing (i.e., what proportion time-dependent designing will improve the travel-time) over static path designing. to deal with the primary argument, note that recent advances in device networks enabled instrumentation of road networks in major cities for aggregation period traffic information, and thus it's currently possible to accurately model the time - dependent travel-times supported the Brobdingnagian quantity of historical information as shown in Fig.1.

B. Time-Dependent Graph

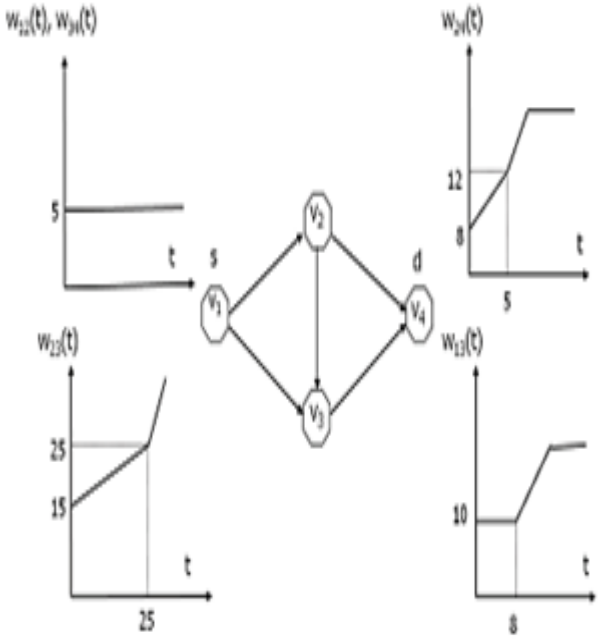

Fig.2. Time-dependent graph.

A special network is formed as Fig.2 a time-dependent graph and edge travel-times area unit perform of your time. take into account the snap of the network (i.e., a static network) with edge weights such as travel-time values at t=0. With classic quickest path computation approaches that disregard time-dependent edge travel-times, the quickest path from s to d goes through v_1, v_2, v_4 with a price of thirteen time units. However, by the time once v_2 is reached (i.e., at t=5), the value of edge $e(v_2, v_4)$ changes from eight to twelve time units, and thence reaching d through v_2 takes seventeen time units rather than thirteen because it was anticipated at t=0. In distinction, if the time-dependency of edge travel-times area unit thought of and thence the trail going through v_3 was taken, the overall travel-cost would are fifteen units that is that the actual best quickest path. We tend to decision this defect of the classic quickest path computation techniques as no-look ahead downside. Sadly, most of the present state o f the art path planning applications (e . g., Google Maps, Bing Maps) suffer from the no-lookahead defect and, hence, their quickest path recommendation remains the same all over the day no matter the departure-time from the supply (i.e., query time).

C. Cell Automata Model

The essential component of a Cellular Automata is the cell. A cell is a sort of a memory component and stores - to say it with simple words - states. In the model embraced in this venture, every cell can have the paired states 1 or 0. In more mind boggling reenactment the cells can have more diverse states. The cells orchestrated in a spatial web frame a cross section. These cells masterminded in a cross section speak to a static state. To bring dynamic into the framework, we need to include rules. The "occupation" of these guidelines is to characterize the condition of the cells for whenever step. In cell automata a standard characterizes the condition of a cell in reliance of the area of the cell. Diverse meanings of neighborhoods are conceivable. Considering a two dimensional grid the accompanying definitions are basic.

- Von Neumann Neighborhood, four cells. The phone above and underneath, right and left from every phone are known as the Von Neumann neighborhood of this cell. The range of this definition is 1, as just the following layer is considered. The aggregate number of neighbor cells including itself is 9 cells
- Moore Neighborhood, eight cells. The Moore neighborhood is a broadening of the von Neumann neighborhood containing the corner to corner cells as well. For this situation, the range r=1 as well. The aggregate number of neighbor cells including itself is 9 cells.
- Extended Moore Neighborhood, proportionate to the portrayal of Moore neighborhood above, however the area comes to over the separation of the following adjoining cells. Subsequently the r=2 (or bigger). The aggregate number of neighbor cells including itself is 25 cells.
- Margolus Neighborhood, a totally diverse methodology: considers 2x2 cells of a grid without a moment's delay. We won't depict Margolus Neighborhood in subtle elements here

IV. TIME-DEPENDENT GRAPH FRAMEWORK

The broadcasting exemplary uses transmission medium like 3G, Mobile WiMAX. once the traffic supplier broadcasts a dataset all driver will hear the dataset at the same time. Thus, this transmission model balances well freelance of the number of driver. Fig.3 the wireless broadcast model the traffic provider repeatedly transmits broadcast cycles, containing the database and air index. the published cycle consists of mounted size packets.

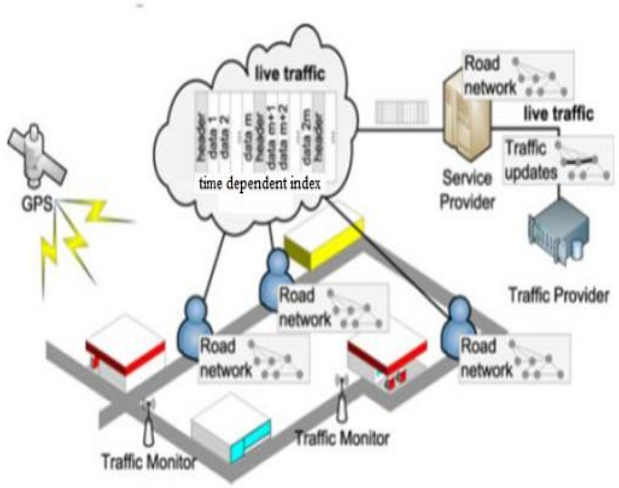

Fig. 3. LTI-TD framework.

The most common wireless broadcasting methodology is the (1, m)interleaving theme, shown in Figure.

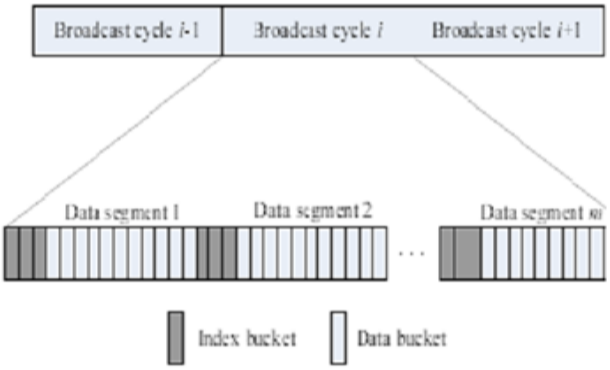

Fig. 4. (1, m) interleaving scheme.

Fig .4 The dataset is split into m distinct segments, and each data section is preceded by the index. this manner the driving force may receive a duplicate of the index instantly when the completion of the presently transmitted knowledge section. A driver will raise algorithmic program one 1st so as to search out the shortest path from a supply to a destination when reading the necessary section, it enumerates the shortest path. In each broadcasting cycle, the driving force 1st collects live traffic updates from the traffic supplier, so updates the graphs. The ALT algorithmic program was projected to search out shortest path on road networks. With ALT, a group of nodes area unit chosen so the shortest path between all the nodes within the network area unit computed. The time-dependent ALT algorithmic program calculates the leaving time from a supply to search out the right path. A driver will raise algorithmic program a pair of so as to search out the shortest path from a supply to a destination. First, the consumer develops a search graph G supported current position and destination. When the driving force keeps paying attention to the published channel till it discovers a necessary section. In order to stay the novelty of LTI-TD, the system is needed to broadcast the latest weight of edges alternatingly.

Algorithm Driver(S: Source; D: Destination) 1:

- Generate G based on s and d.
- Listen to the channel for a segment.
- Decide the necessary segments.
- Compute the shortest path (from s to d) on G.

In the above algorithm driver generate a graph and listen the list of channels then find the necessary segment. And to find out the computing shortest path.

Algorithm Traffic-Provider (G:graph) 2:

- construct G.
- for each broadcast cycle do.
- collect traffic updates from the traffic provider.
- update the graphs G.
- broadcast the graph G.

In the above algorithm traffic provider construct a graph for each broadcast cycle. Then collect traffic updates from the traffic provider and update the broadcast graph G.

A. Cell Automata Rule Application

The sort of guidelines utilized in this task is a purported "Totalistic" Rule. That is, the condition of the following state center cell is just needy upon the entirety of the conditions of the area cells, and its own present state. Every dead cell has state "0" as worth, and each alive cell has state "1". We compute the entirety of the conditions of all the contiguous and corner to corner neighbors cells. At the end of the day, in Fig. 1 (b) for occasion all the blue cells are neighbors of the red focal cell, in addition to itself. Subsequently the aggregate number of neighborhood wach cell has is equivalent to 9. In this way the aggregate of the alive neighbor cells can be at most extreme 9*1=9 (all the neighbor cells are alive), and at the base 0*1=0 (all the neighbor cells are dead).

- Number of states: 2. They are "1" alive, and "0" dead.
- Number of neighbors: 10. The quantity of alive neighbors can be from 0 to 9.
- Subsequently, NDR = 2 to power 10 = 1024. (Aggregate of 1024 examples).

This is an entirely vast number of conceivable tenets to be tried. It is worth to understand that not every one of the standards are fascinating, with respect to case on the off chance that we pick a guideline that murders every one of the cells notwithstanding the quantity of alive neighbors, or the inverse, a principle that keep every one of the cells conceived paying little mind to the quantity of alive neighbors. It is intriguing to choose just the guidelines that present more productivity for edge distinguishing of any sort of CA cell.

V. GENERALIZATION TECHNIQUES

The time-dependent quickest path downside (in inventory accounting networks) will be solved by modifying Dijkstra rule. we have a tendency to sit down with changed Dijkstra rule as time-dependent Dijkstra (TD-Dijkstra). TD-Dijkstra

visits all network nodes accessible from s in each direction till destination node d is reached. On the opposite hand, a time dependent A* rule will considerably cut back the amount of nodes that have to be compelled to be traversed in TD-Dijkstra rule by using a heuristic perform h (v) that directs the search towards destination. to ensure best results, h (v) should be admissible and consistent (a.k.a, monotonic). The acceptability implies that h (v) should be but or up to the particular distance between v and d. With static road networks wherever the length of a foothold is constant, euclidean distance between v and d is employed as h (v). However, this straightforward heuristic perform can't be directly enforced to time-dependent road networks, because, the best travel-time between v and d changes supported the departure-time tv from v. Therefore, in time-dependent road networks, we want to use a calculator that ne'er overestimates the travel-time between v associated d for any attainable tv. One straight forward lower-bound calculator is deuc (v, d)/max (speed), i.e., the geometer distance between v and d divided by the most speed among the sides within the entire network. though this calculator is absolute to be a lower-bound, it's a awfully loose bound, and therefore yields insignificant pruning. With our approach, we have a tendency to get a way tighter sure by utilizing the pre-computed distance labels. Presumptuous that associate on-line time-dependent quickest path question requests a path from supply s in partition Si to destination d in partition S_j, the quickest path should pass through from one border node B_i in S_i and one more border node b_j in S_j. We know that the time-dependent quickest path distance passing from Bi and b_j is bigger than or equal to the pre-computed lower-bound border-to-border (e.g., LT T (b_l, b_t)) distance for S_i and S_j combine. We have a tendency to conjointly recognize that a time-dependent quickest path distance from s to Bi is often bigger than or up to the pre-computed lower-bound quickest path distance of s to its nearest border node BS.

VI. EXPERIMENTAL RESULTS

The travel data set Contains source location, Destination location and Time to travel from source to destination.

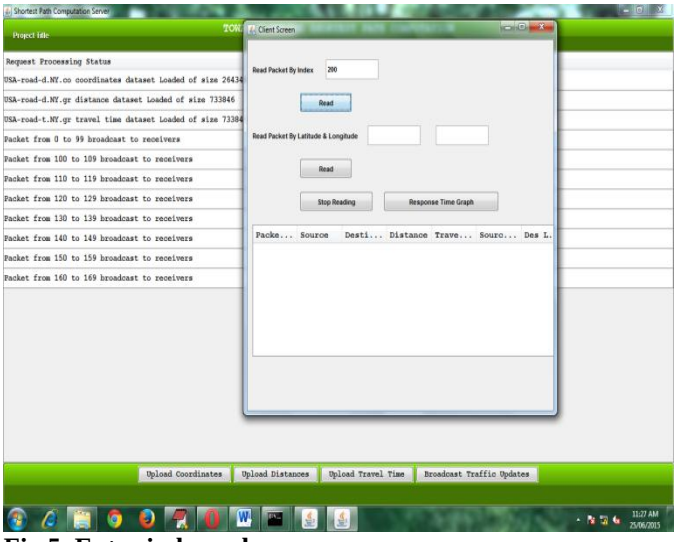

Fig.5. Enter index value.

Fig5 shows the Enter Index value (Location number) then, we can read through latitude and longitude values: Here we have to enter source latitude and longitude values and Destination latitude and longitude values.

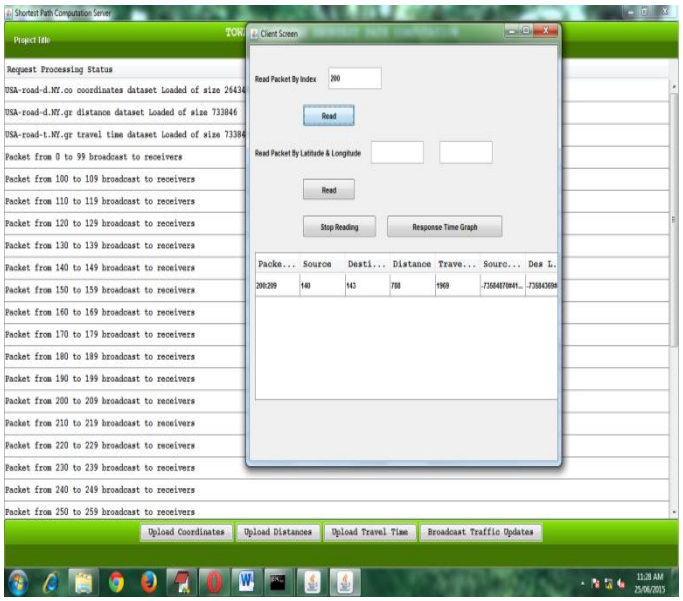

Fig.6. Read values.

Fig.6 shows Click the Read values then we can find out the read values are shown in the table.

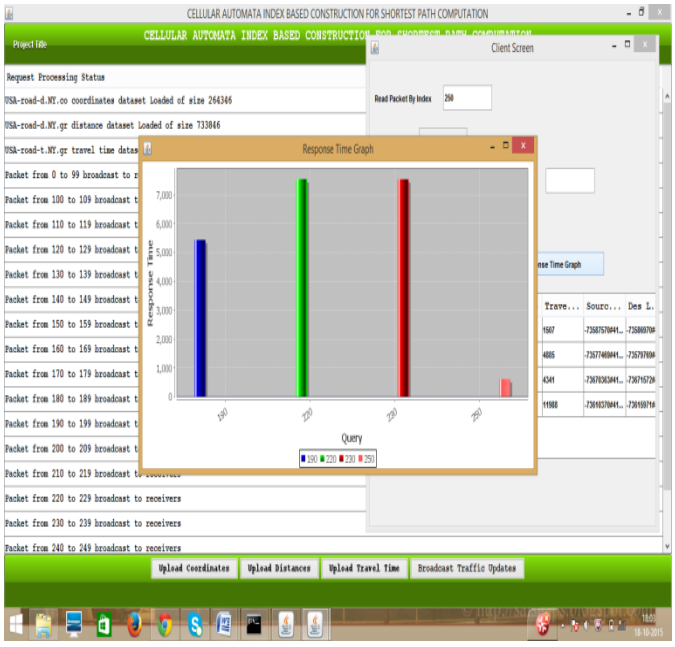

Fig.7. Response time graph.

Fig.7 Our experimental study shows that LTI is strong to varied parameters and it offers comparatively short tune-in value (at consumer side), quick question latent period (at consumer side), tiny broadcast size (at server side), and light-weight maintenance time (by server side) for shortest path downside.

VII. CONCLUSION

To address the matter of economical shortest path in trendy navigation systems within the presence of varied speed conditions on an outsized scale road network, a promising architecture that broadcasts the index on the air reckoning on time is needed. The existing systems were unfeasible to unravel the problem because of their preventative maintenance time and large transmission overhead. LTI-TD may be a novel answer for Online Shortest Path Computation on Time Dependent Network.

Future Work: Our future work will extend solution on time dependent networks. This is a very interesting topic since the decision of a shortest path depends not only on current traffic data but also based on the predicted traffic circumstances.

VIII. REFERENCES

[1] Leong Hou U, Hong Jun Zhao, Man Lung Yiu, Yuhong Li,and ZhiguoGong"Towards Online Shortest Path Computation" IEEE Transactionson Knowledge And Data Engineering,Vol. 26,No. 4,April 2014.

[2] Hector Gonzalez, Jiawei Han, Xiaolei Li, Margaret Myslinska, John Paul Sondag"Adaptive Fastest Path Computation on a Road Network: A Traffic Mining Approach" VLDB „07, September 23-28, 2007, Vienna, Austria.Copyright 2007 VLDB Endowment, ACM 978-1 - 59593-649-3/07/09.

[3]UgurDemiryurek,FarnoushBanaei-Kashani,CyrusShahabi, andAnandRanganathan" Online Computation of Fastest Path in Time-Dependent Spatial Networks" D. Pfoser et al. (Eds.): SSTD 2011, LNCS 6849, pp. 92–111, 2011.c SpringerVerlag Berlin Heidelberg 2011.

[4] KyriakosMouratidis "Spatial Queries in Wireless Broadcast Environments"MobiDE'12, May 20, 2012 Scottsdale, Arizona, USA.

[5] Fabian Fuchs "On Preprocessing the ALTAlgorithm" Karlsruhe, den 12. Juli 2010 Ort, Datum.

[6] Batz, G.V., Delling, D., Sanders, P., Vetter, C.: Time-dependent contraction hierarchies. In: ALENEX (2009).

[7] Cooke, L., Halsey, E.: The shortest route through a network with timedependent intermodal transit times. Journal of Mathematical Analysis and Applications (1966).

[8] Dean, B.C.: Algorithms for min-cost paths in time-dependent networks with wait policies. Networks (2004).

[9] Dehne, F., Omran, M.T., Sack, J.-R.: Shortest paths in time-dependent fifo networks using edge load forecasts. In: IWCTS(2009).

[10] Delling, D.: Time-dependent SHARC-routing. In: Halperin, D., Mehlhorn, K. (eds.) Esa 2008. LNCS, vol. 5193, pp. 332–343. Springer, Heidelberg (2008).

[11] Sree, P. Kiran, I. Ramesh Babu, and NSSSN Usha Devi. "Investigating an Artificial Immune System to strengthen protein structure prediction and protein coding region identification using the Cellular Automata classifier." International journal of bioinformatics research and applications 5, no. 6 (2009): 647-662.

[11] Sree, P. Kiran, and I. Ramesh Babu. "Face Detection from still and Video Images using Unsupervised Cellular Automata with K means clustering algorithm." arXiv preprint arXiv:1312.6834 (2013).

[12] Sree, P. Kiran, I. Ramesh Babu, J. V. R. Murty, R. Ramachandran, and NSSSN Usha Devi. "Power-aware hybrid intrusion detection system (PHIDS) using cellular automata in wireless ad hoc networks." WSEAS Transactions on Computers 7, no. 11 (2008): 1848-1874.