**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**

***IJCSMC, Vol. 4, Issue. 6, June 2015, pg.1147 – 1153***

**RESEARCH ARTICLE**

# Online Direction Computation Using Current Traffic Index

**Soniya Rajan[1], Shubhangi D.C[2]**

[1]Department of Computer Science and Engineering VTU, RO Kalaburugi, India

[2]Department of Computer Science and Engineering VTU, RO Kalaburugi, India

[1] soniarajan.enathu@gmail.com; [2] shubhangidc@yahoo.co.in

***Abstract— Online direction computing aims to find the shortest path using current traffic circumstances. Shortest path computation is one of the most common queries in location-based services. Shortest path problems can be solved very efficiently when a directed graph is nearly acyclic. Earlier results defined graph decomposition, now called the 1-dominator set, which consists of a unique collection of acyclic structure dominated by a single associated trigger vertex. In this framework, a specialized shortest path algorithm only spends delete-min operations on trigger vertices, thereby making the computation of shortest paths through non-trigger vertices easier. The network shortest pathway problem desire at computing the precise path placed on live traffic chances. Beyond live traffic classes, the route returned by the navigation system is no longer insured and definite result. A promising access is let the server to gather live traffic data and then newscast them over radio or wireless network. There exist many methods but there is no efficient system/solution that can offer affordable costs at both client and server sides for online shortest path computation. Unfortunately, the conventional client-server architecture scales poorly with the number of clients. Thus, we develop a new framework called current traffic index (CTI) which enables drives to quickly and efficiently collect the current traffic information.***

***Keywords— "index, shortest path, broadcasting"***

## I. INTRODUCTION

Data mining is an interdisciplinary subfield of computer science is the computational process of discovering patterns in large data sets involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into and understandable structure for further use. Shortest path computation is an important function in modern car navigation systems. Typical client-server architecture can be used to answer shortest path queries on live traffic data. In this case, the navigation system typically sends the shortest path query to the service provider and waits the result back from the provider (called result transmission model). However, given the rapid growth of mobile devices and services, this model is facing scalability limitations in terms of network bandwidth and server loading.

Greater productivity, more conventional communication and everyday need for data-on-demand are just some of the reasons that mobile devices are becoming increasingly popular. Currently we are using many services which provide current traffic information by considering snapshots of the area based on the queries. But they cannot report continuously due to the operating costs. So for answering this problem here we are using Current Traffic Index (CTI).

Normally used client-server architecture for finding the shortest path is facing scalability problem in terms of network bandwidth and server loading. The scalability in terms of number of clients and the amount of current traffic updates are the main challenge while finding the shortest path while considering the current traffic.

The rest of the paper is organized as follows. Section II reviews the existing approaches. Section III describes the system overview of CTI. Section IV provide analytical and simulation results. Section V draws the conclusion.

## II. RELATED WORK

The efficient shortest path query answering on graph [1] using TEDI (TreE Decomposition based Indexing), an indexing and query processing scheme for the shortest path query answering. And it introduced scheme based on the tree decomposition concept for the shortest path query answering. TEDI is based on the tree decomposition methodology. By using sparse graphs we can attain good performance but the compression methods with the information loss cannot be adapted for answering shortest path queries and it cannot present the query time for the synthetic data, we cannot make a comparison on that. TEDI achieves the improvement of performance by more than an order of magnitude in all aspects including query time, index construction time and index size, but cannot able to maintain the index structure.

The reductions are achieved by introducing a construct termed a distance oracle that yields an estimate of the network distance between any two vertices in the spatial network [2]. But here approximate network distances cannot provide a reasonable answer for query processing as the resulting error.

The data are continuously transmitted on the air, while clients listen to the broadcast and process their queries locally. Wireless broadcasting has been considered for spatial processing in Euclidean space [3]. However, there is currently no work no road networks. They empirically demonstrate the efficiency of our techniques using real networks and actual device specifications. Broadcast model is that it can support an arbitrary number of users/queries, since no processing takes place at the server and the network overhead is irrelevant to the number of clients. But the disadvantages are the memory requirements are a major concern in our problem, yielding several methods inapplicable to large or even moderately sized road networks.

N. Malviya, et al. [4] proposed two classes of approximate techniques-K-paths and proximity measures to substantially speed up processing. The set of designated routes specified by continuous route planning queries in a face of incoming traffic delay updates. By using ellipse is that it restricts the number of segments that need to consider while processing updates that we might otherwise consider unnecessarily. Continuous monitoring of all routes has the advantage that the user is always of the best route. The disadvantage is it do not help us to scalability issues in using them at run time for a large set of registered routes. The proximity recomputes the best shortest path every time it issues an update and there is always optimal in its suggested cost.

L. Wu, X. et al. presented a comprehensive comparison of the most advanced spatial-coherence-based and vertex-importance-based approaches. The performance of the state-of-the-art spatial-coherence based algorithms; SILC and PCPD were tested using only small road networks with up to one hundred thousand vertices. PCPD in inferior to SILC in terms of pre-computation cost, space consumption, as well as query efficiency. The major disadvantages are computing the shortest path between two locations in a road network is an important problem that finds applications in various map services and commercial navigation products. This algorithm is simple and elegant, but it is often inefficient for sizeable road networks.

We are proposing a road network monitoring system typically consists of a service provider, a large number of mobile clients and a traffic provider shows an architectural overview of this system in the context of our live traffic index framework. The traffic provider collects the live traffic circumstances from the traffic monitors via techniques like road sensors and traffic video analysis. In order to keep freshness of the broadcast index, the cost of index maintenance is necessarily minimized. Mobile client hope to compute and monitor a shortest path, it tune in to the live traffic index and reads the important part of the index for calculating the shortest path. Here we are achieving the fast query response time, limited tune-in-cost, small broadcast size and light maintenance time.

## III.METHODOLOGY

A road network monitoring system consists of server provider, traffic providers and number of clients. Vehicles are an example for the clients and we will get traffic details from the GoogleMap, TEQ,etc.

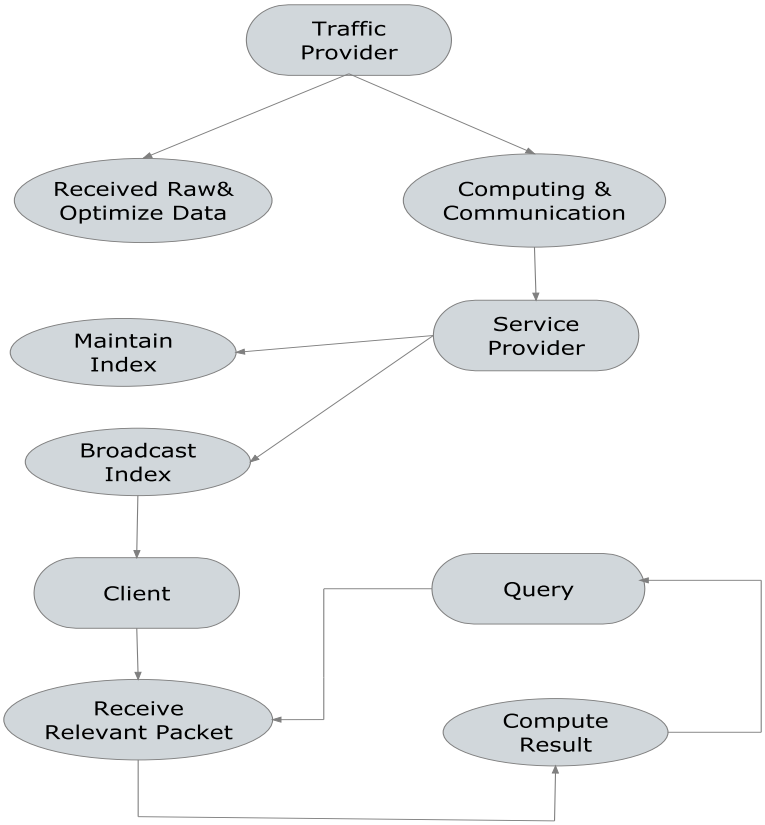



Fig. 1. System Architecture

In Fig. 1 shows the architecture the overview of this system. The traffic provider collects the current traffic information. The traffic monitors collects information through many techniques like traffic video analysis and road sensors. The service provider periodically receives current traffic updates from the traffic provider and broadcasts the current traffic index on radio or wireless network. The client listens to the current traffic index from the server provider and read relevant portion of the index and computes the shortest path. In order to provide the current traffic information the server maintains and broadcasts index according to the current traffic data. In order to compute the online shortest path a client listens to the index and computes the shortest path.

Real-time traffic feeds are collected from the traffic monitoring resources planted along the roadside and other crowd-sourcing techniques by the provider. The collected information is shared to different service provider both as raw and optimized data. We are providing the live feed to the service providers which are collected from the advanced real-time traffic simulator. This channel provides a preprocessed traffic data to the Service provider.

The preprocessed raw traffic data collected from the providers are being indexed and are orderly broadcasted towards multiple clients. The traffic datum is organized by indexing the collected information such as to provide real-time feeds to the clients, and also in the manner to ease the client with their computation cost.

The clients which are in need of the real-time traffic feeds are tuned in a manner to receive the relevant data packets based on the indexed traffic data. The received packets are then used for computing the shortest path query, q(s,d). Generally this computation part is carried over in the server but here we do it in the client part to reduce computation overload for the server and to reduce significant computation cost.

The main objectives of the CTI are:

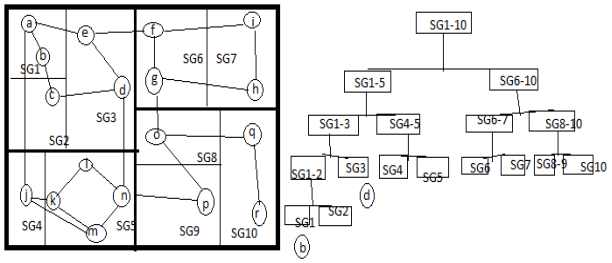
- It reduces the maintenance time spent at maintain index.
- It can save the communication cost at broadcast index and receiving relevant packets by maintain the index size in a reasonable ratio.
- Efficiency computation on a portion of entire index enables clients to compute shortest path on a portion of the entire index. This reduces the amount of data received.

### *A. CTI Construction*

Hierarchical index structures enable fast shortest path computation. It helps to find it on a portion of the entire index which helps to reduce the tune-in-cost at the client while transmitting.

Given a road network graph G =(VG; EG) index structures partitions G into a set of small sub-graphs SGi and organizes SGi in a hierarchical fashion (i.e., tree). In Fig. 2, we illustrate a graph being partitioned into 10 sub-graphs (SG1, SG2; . . . ; SG10) and the corresponding hierarchical index structure. Every leaf entry in a hierarchical structure represents a sub-graph SGi that consists of the corresponding nodes and edges from the original graph. For instance, SG1 consists of two nodes VSG1 ={a, b} and one edge ESG1 ={ a, b}. A non-leaf entry stores the inter-connectivity information between the child entries. Shortcuts (Δ_( 〖SG〗_i ) ) are the common type of pre-computed information in these index entries. A common approach is tofetch the relevant entries from the index using the bottom-up execution fashion.




a. Road ntework b. hierarchical tree view

Fig. 2. Hierarchical index structure

*Cost analysis*. The total space requirement of hierarchical index can represented as follows.

$$|I| = \sum_{SG_i \in I}\left(|V_{SG_i}| + |E_{SG_i}| + |\Gamma_{SG_i}| + |\Delta_{SG_i}|\right) + tree \quad (1)$$

where $V_{SG_i}$ and $E_{SG_i}$ represent the nodes and edges in $SG_i$, respectively, $\Gamma_{SG_i}$ represents the connectively information between the child entries, $\Delta_{SG_i}$ represents the pre-computed information of I.

The space requirement can be revised as the follows:

$$|I| \approx |G| + \sum_{SG_i \in I}|\Delta_{SG_i}|. \quad (2)$$

To minimize the index broadcast size, it is more or less equivalent to minimize the size of $\Delta_{SG_i}$.

*B. Index construction*

One possible solution is to relax the optimization objectives which makes them be the tunable factors of the problem. While the overhead of pre-computed information and the number of relevant entries cannot be decided straightforwardly, we decide to relax the first objective (i.e., minimizing the size of leaf entries) such that it becomes a tunable factor in constructing the index. Our objective to find a hierarchical index structure I such that,

$$OBJ(I) = \min_{SG_i \in I} \frac{\Sigma|\Delta_{SG_i}|}{min\{|V_{SG_i}|\}} \quad (3)$$

where $min\{|V_{SG_i}|\}$ can be viewed as a normalized factor such that the objective function prefers balanced partitions.

*Conditional Partitioning Algorithm:*

Algorithm 1 Conditional Partitioning Algorithm

PQ: a priority queue; I: index structure;

Algorithm partition(G: the graph, γ: the number of ;partitions)

1. ( λ,ν):= *eign*(G) and n :=root of *I*
2. Insert (n,G, ν, λ) intoPQ in decreasing order to
3. While $|PQ| < \gamma$ do
4. (n,G, ν, λ) := PQ.*pop*()

5. For *k* :=2 to $\gamma - |PQ| + 1$ do
6. decompose G into $SG_1 \ldots\ldots\ldots\ldots SG_k$
7. form a temporal index $I'$ that attaches $SG_1..SG_k$
8. if avg(S($I'$)) is better than $best_{SG}$ then
9. update $best_{SG}$ and $best_{SG}$ :={ $SG_1..SG_k$}
10. attach $best_{SG}$ as $n's$ children
11. for i:=1 to $|best_{SG}|$ do
12. insert ($n_i, SG_i, \nu_i, \lambda_i$) into PQ
13. return *I*

### *C. CTI Transmission*

We first partition the index structure and the weight of the edges to broadcast a hierarchical index (l,m) interleaving schema[6] . it is an index allocation method in which the index is broadcast m times during the one version of the file. We partition the index into two parts: the index structure and weight first in order to broadcast the index using the (l, m) interleaving scheme. In order to keep the freshness of CTI, our system is required to broadcast the latest weight of edges periodically.

| offset | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | id | | | | Checksum | | | | | | | | | | | |
| …… | | | | | | | | | | | | | | | | |

Table: 1 Packet format on the Air Index.

Table. 1 shows the format of a header /data packet in our model. *id* is the offset of the packet in the present broadcast cycle and *checksum* is used for error-checking of the header and data.

## IV. RESULTS AND DICUSSIONS

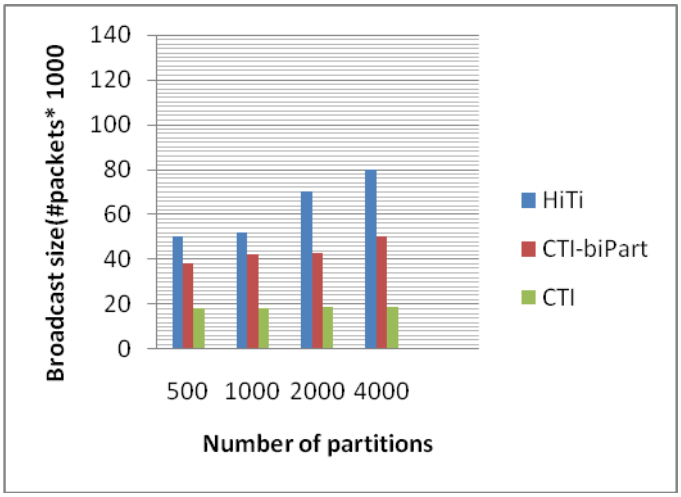
| City | | Method | | | | |
|---|---|---|---|---|---|---|
| | | Raw BD | Transmission DALT | Index CH | Transmission HiTi | Model CTI |
| NYC | T | 17300 | 17320 | 17300.7 | 26825.2 | 607.7 |
| | R | 64.4 | 44.3 | 1.28 | 154.11 | 4.26 |
| | B | 22930 | 24830 | 124870 | 12487 | 35611 |
| | M | - | - | 1579.6 | 105454 | 5675.4 |
| SJ | T | 2291.2 | 2291.2 | 725.0 | 1524.1 | 331.1 |
| | R | 5.2 | 1.98 | 0.08 | 9.72 | 1.37 |
| | B | 353 | 3535 | 1280 | 5603 | 3000 |
| | M | - | - | 262 | 635 | 86.6 |
| OB | T | 322 | 322 | 250 | 619 | 0.88 |
| | R | 1.1 | 0.35 | 22.2 | 4.66 | 0.55 |
| | B | 504 | 504 | 48 | 1336 | 777 |
| | M | - | - | 102 | 102 | 14 |

Table 2 Performance of different methods
Client side: **T**une-in-cost (#packets), **R**esponse time (ms).
Server side: **B**roadcast size (#packets), **M**aintenance time (ms).

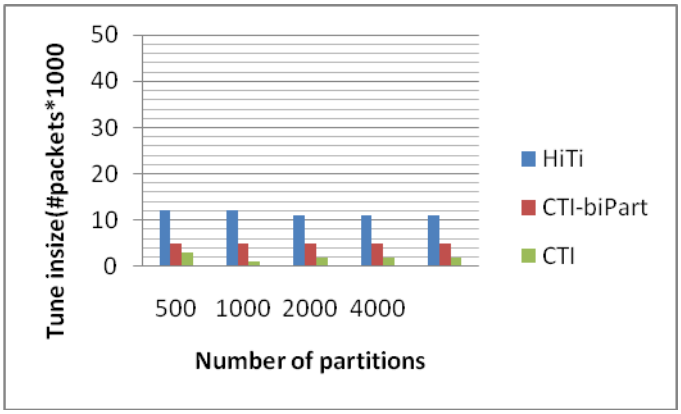
We test four different road maps, including New York (NYC-264k nodes, 733k edges), San Joaquin road map (SJ-174k nodes, 443k edges), Oldenburg road map (OB-6k nodes, 14k edges) from [7], [8]. For each method,

we measure its performance in term of its tune-in-cost, maintenance time, broadcast time and query response time.




(a) Service provider




(b) Client

Fig. 3. Varying number of partitions. γ.

In fig. 5 plots performance of all three methods as a function of the number of partitions , γ on the data set. For the sake of saving space, we plot the costs at service provider (i.e., broadcast size and maintenance time) into one figure and plot the costs at client (i.e., tune-in size and response time) into another figure. The number of packets is by bars (left y axis.

## V. CONCLUSIONS

The shortest path result is computed based on the current traffic information. We carefully analyze the existing work and discuss their inapplicability. The main existing problem is the maintenance of the index structure and takes large transmission overhead. To address the problem, we suggest a promising architecture Current Traffic Index (CTI). CTI can reduce the maintenance time spent at the server provider. And by using the hierarchical index structure enables client to calculate the shortest path on a small portion of the index. CTI satisfies all aspects- fast query response time, limited tune-in-cost, small broadcast size and light maintenance time. In the future, we will extend our CTI solution on time dependent networks.

## REFERENCES


[1] S F. Wei, "TEDI: Efficient shortest Path Query Answering onGraphs,"Proc. VLDB Endowment, vol. 3, no. 1, pp. 741-757,2010.

[2] J. Sankarannarayanan and H. Samet, "Query Processing Using Distance Oracles for Spatial Networks," IEEE Trans. Knowledge and Data Eng., vol. 22, no. 8, pp. 1158-1175, Aug.2010.

[3] G. Kellaris and K. Mouratidis, "Shortest Path Computation onAir Indexes," Proc. VLDB Endowment, vol. 3, no.1, pp. 741-757, 2010.

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

[5] L. Wu, X. Xiao, D. Deng, G. Cong, A.D. Zhu, and S. Zhou, "Shortest Path and Distance Queries on Road Networks: An Experimental Evaluation," Proc. VLDB Endowment, vol. 5, no. 5, pp. 406-417, 2012.

[6] T. Imielinski, S. Viswanathan, and B.R. Badrinath, "Data on Air: Organization and Access," IEEE Trans. Knowledge and Data Eng., vol. 9, no. 3, pp. 353-372, May/June 1997.

[7] "9th DIMACS Implementation Challenge-Shortest Paths," http:// www.dis.uniroma1.it/challenge9/download.shtml, 2014.

[8] "Network-Based Generator of Moving Objects," http://iapg.jade-hs.de/personen/brinkhoff/generator/, 2014.

[9] "NAVTEQ Maps and traffic,"http://www.navteq.com,2014.

[10] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2010-2015,"2011.chical and Goal-Directed Speed.

[11] R. Bauer, D. Delling, P. Sanders, D.Schieferdecker, D. Schultes, and D.Wager, "Combining Hierarchical and Goal-Directed Speed-Up Techniques for Dijkstra's Algorithm,"ACMJ.ExperimentalAlgorithmics,vol.15,article 2.3, 2010.

[12] T. Beuhler and M. Hein, "Spectral Clustering Based on The Graph -Laplacian," Proc. Int'l Conf. Machine Learning (ICML), p. 11, 2009.

[13] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 888-905, Aug. 2000.

[14] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, second ed. Morgan Kaufmann, 2006.

[15] "9thDIMACS Implementation Challenge-Shortest Paths," http: //www.dis.uniroma1.it/challenge9/download.shtml, 2014.

[16] L. Wu, X. Xiao, D. Deng, G. Cong, A.D. Zhu, and S. Zhou, "Shortest Path and Distance Queries on Road Networks: An Experimental Evaluation," Proc. VLDB Endowment, vol. 5, no. 5, 406-417, 2012.

[17] R. Bauer and D. Delling, "SHARC: Fast and Robust Unidirectional Routing," Proc. 10th Workshop Algorithm Eng. and Experiments (ALENEX), pp. 13-26, 2008.

[18] T. Buhler€ and M. Hein, "Spectral Clustering Based on The Graph - Laplacian," Proc. Int'l Conf. Machine Learning (ICML), p. 11, 2009.

[19] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 8, 888-905, Aug. 2000.