

Dynamic Balancing of Push and Pull in a Distributed Traffic Information System

Qingfeng Huang and Ying Zhang

Palo Alto Research Center Inc., 3333 Coyote Hill Rd, Palo Alto, CA 94304, USA

Emails: {qhuang, yzhang}@parc.com

Abstract—In this paper, we propose and study a novel distributed vehicle traffic information system using a wireless sensor network. Besides the system model, the contributions of the paper include a road network aware information publication protocol, a distributed query processing protocol with transient memory, and the adaptive interaction model between the two, all in the context of a vehicle traffic information system. Both theoretical and simulation results are presented.

I. INTRODUCTION

Improving transportation safety, mobility, and efficiency are the key goals of intelligent transportation system (ITS). With the advance of communications and electronics technologies, the vision of ITS is becoming more and more approachable, and research in this area is gaining momentum in recent years. Real-time traffic information system (RTIS) is one of the key enabling components in the ITS vision, by providing vital traffic flow information (e.g. flow speed and congestion level) for intelligent arterial management, freeway management, transit management, traveler information system, real-time path planning, etc. A few centralized traffic information system concepts [1][2][3][4] have been developed, tested, and evaluated by both the public and the private sectors. While a reasonable starting point for testing, the centralized architecture will eventually face the classic scalability problem when users grows.

In this paper we introduce a distributed ad hoc traffic information architecture using a wireless sensor network. A unique perspective of the sensor network’s application in the traffic information system domain is that the direction and scope of its information dissemination on the network could be optimized using the knowledge of the road and traffic network. To take advantage of this key observation, we introduce a road-network-aware information publication protocol, a distributed query processing protocol with transient memory, and a model of distributed adaptive interaction between the two for efficient distributed information dissemination and discovery for vehicle traffic information applications. Both theoretical model and simulation results are presented.

The remainder of paper is organized as follows. Section II introduces the abstract system model our study is based on. Section III describes our information publication and query protocols. Section IV presents some theoretical results. Section V shows the simulation results, followed by conclusion Section VI.

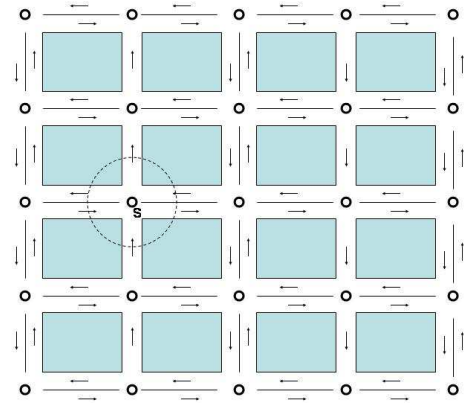


Fig. 1. Road configuration with traffic sensors in the intersections.

II. SYSTEM MODEL

We model the distributed traffic information system using two networks, a *road/traffic network* and a *sensor communication network*. Fig. 1 shows a simple example of a road configuration with traffic sensors in the road intersections. The arrows indicate vehicle traffic directions, and the little circles indicate the positions of traffic sensors. The dashed circle in the figure indicates the sensing area of the corresponding sensor S .

A *road traffic network* is defined by nodes, links and connectivity between links. A *node* is a point with an X-Y-Z location coordinate. A *link* is a directional road segment defined by two nodes $L(n_1, n_2)$, where n_1 is the start and n_2 is the end node of the link. An abstract road network is defined to be the directed graph on links. A link L_1 connects to L_2 , denoted $L_1 \rightarrow L_2$, if and only if the end node of L_1 is the start node of L_2 and vehicles in L_1 are allowed to move to L_2 .

For clarity, Fig. 2 only shows a simple example of four intersections (Fig. 2(a)) like in the top-left corner of Fig. 1. Each intersection is represented by a node. The link connectivity graph is shown in Fig. 2(b), where for instance, $L(1, 2)$ denotes link from intersection 1 to intersection 2, and $L(2, 1)$ denotes a link from intersection 2 to intersection 1. Note that there is no directed connection from $L(3, 4)$ to $L(4, 3)$ although the end node of $L(3, 4)$ is the start node of $L(4, 3)$; they would be connected only if a U-turn is allowed between these two segments at the intersection 4. The road

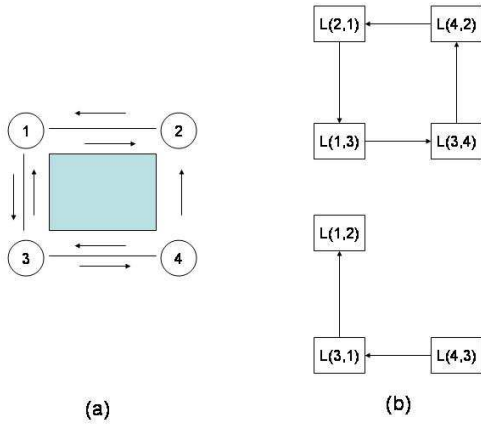


Fig. 2. Road/traffic network abstraction.

network abstraction is useful in determining how “relevant” an information about a link is to vehicles at another link. The ability to make this determination is in turn useful for deciding effective information dissemination and query scopes.

Fig. 3 shows the sensor network abstraction (an undirected graph with each link indicating the existence of a wireless network connection) of Fig. 1, and we assume a sensor node is located at a node of the traffic network. The solid

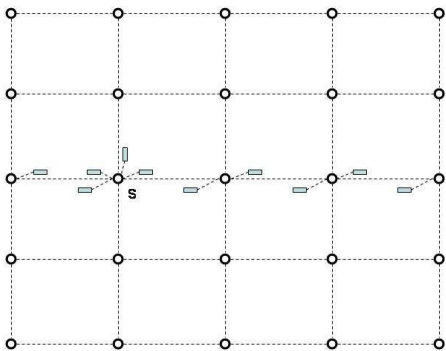


Fig. 3. Sensor network abstraction.

rectangles nodes in Fig. 3 are examples of nomadic client nodes of the traffic sensor network. For simplicity, our model does not consider ad hoc vehicle-to-vehicle communication for information exchange. We assume the radio range of the wireless node to be fixed, and radio range is large than the longest link length so that the sensor network is connected. The connectivity of both road and sensor networks are stable during the operation of the protocol.

In our distributed architecture model, traffic information are stored distributedly in the sensor network for access by end users instead of going through a central server [5]. We allow each vehicle in a link to query road traffic information about the links it may travel in the future. And each sensor node stores link information of the link that ends at this location

as well as link information propagated by neighboring nodes. For example, node 1 in Fig. 3(a) stores information for $L(2, 1)$ and $L(3, 1)$, in addition, it may have information on $L(1, 3)$ and $L(3, 4)$, if nodes 3 and 4 propagates the information or a vehicle on link $L(2, 1)$ requests such information.

Link information can be sensed or computed by infrastructure sensors installed in each road segment/intersection or by communication. For example, a wide angle video camera at the intersection can detect congestion; multiple magnetic sensors in each road segment can collectively detects average speed of the traffic along the link. Furthermore, a sensor may be aware of traffic light situations in the intersection so that it can do traffic flow speed estimation that takes into account the traffic light caused traffic stoppage. The duration of a vehicle traveling along the link may be recorded to obtain the speed information along the link. The abstract sensor in our model represents the collective of these types. We also assume each sensor node has information about its location and a local road traffic network. Although we do not assume the client (vehicle) nodes have GPS information, in most cases, our system may be used with a GPS road navigator for path planning and real-time navigation guidance.

To save the bandwidth of the communication it is important to reduce the total amount of transmissions for information sharing. Consider the characteristics of traffic information applications, link information is propagated along the road network to the nodes that are likely to be queried on the information. The rest of the paper will discuss how to optimally define the boundary of the information propagation based on frequencies of queries.

III. INFORMATION PROTOCOLS

Balancing push and pull is a key notion in optimizing a distributed information system. For a distributed traffic information system, if the traffic information about a specific road segment is only stored on the node that senses the information, then a client that wants to know the traffic condition about the road segment needs to send a query all the way to the source. This is fine when the request rate compared to the generation rate of information is relatively low. However, when the request rate exceeds certain threshold, it will be more efficient to push the information to a certain scope in anticipating the demand of information [6].

For push, we introduce a new information publication (push) protocol that works on road network topology rather than the geographical topology. For pull, we develop a distributed query (pull) processing protocol with transient memory for increasing distributed query efficiency. Furthermore, the push and pull protocol interact with each other using actual information demand rates and information generation rates to further increase efficiency.

A. Road Network Aware Publication Protocol

A simple form of the road network aware information publication protocol is specified by:

$$\langle m, R \rangle \quad (1)$$

where R defines the recipient set based on a road network topology constraint, m is a message that can be further defined as a 4-tuple:

$$m = \langle l, i, t, \tau \rangle \quad (2)$$

where

- l is the link ID,
- i is the link information, such as average speed, congestion status, etc.
- t is the time that the link information is obtained,
- τ is the expiration time.

One possible constraint of R is the push scope constraint indicating how far a link information should be pushed along the road network. For example, if $R = T$, then a link information shall be pushed to the nodes that are at most T seconds away (given current, historical or nominal vehicle speed information on relevant links) from the start node of the link. This way a vehicle that is T seconds away from the start of the link can access the link information locally. If the published information is *event-driven*, then $\tau = \text{inf}$, which means the information will not expire until next publication.

B. Query Processing Protocol

A query in our design is specified by a pair:

$$\langle n, \mathcal{L} \rangle \quad (3)$$

where

- n is the ID of the node generating the query,
- \mathcal{L} is a list of $\langle l, c \rangle$ pairs, where
 - l is the ID of the link that is queried,
 - c is a constraint on the information of interest on the respective link, (e.g, how “fresh” the information needs to be).

C. Push and Pull Implementation

1) *Push*: In our prototype implementation, each sensor node stores a list of link information and its status. Each entry of the list includes a 4-tuple

$$\langle l, i, t, \tau \rangle \quad (4)$$

where l, i, t, τ are link ID, link information, the time when the information is generated and expiration time, respectively.

The link information is published by the sensor node located at the end of the link either periodically or on detection of a significant event, i.e., when the value change exceeds certain threshold. When a sensor node n receives a new piece of link information $\langle l, i, t, \tau, R \rangle$, it checks its constraints in R : if R is not satisfied, or it is expired (i.e., $t + \tau$ is smaller than the current time), then the message is dropped, otherwise:

- if l does not exist in the list, add the entry with $\langle l, i, t, \tau \rangle$,
- if l exists in the list with smaller t , update the entry with $\langle l, i, t, \tau \rangle$ and rebroadcast $\langle l, i, t, \tau, R \rangle$.

Note that if the recipient set R and the source are self-connected without other relay nodes, and communication links are reliable, then this simple implementation would guarantee that all recipients within the expiration time bound will receive the information.

2) *Pull*: For simplicity, we generate one query for each link in \mathcal{L} . When a network node receives a query $\langle n, l, c \rangle$ from a vehicle, it checks to see if it has relevant data about link l that satisfies the constraint c in the link list.

If the information does not exist locally, the query will be propagated along the shortest communication path from this node to the destination node, i.e., the end node of link l which definitely has the information. Along the way to its destination, as soon as the query hits a node that has the piece of information, a reply $\langle l, i, \min(t, t + \tau), 0 \rangle$, where t is the current time, is generated and propagated along the shortest path from this node to the origin of the query. Each node that receives or overhears the reply will update its link list with the information.

Note that using shortest path routing is just a simple implementation of this algorithm. If the routing table is not built a priori, one can have ad-hoc routing such as greedy geo-forwarding [7].

A more efficient implementation is to generate one query for all links that are queried, which may share some of the query propagation and reply. If a node has information about a subset of links \mathcal{L}' in \mathcal{L} , it sends a reply back to n with relevant information about \mathcal{L}' . A multicast tree for passing the rest of query $\mathcal{L} - \mathcal{L}'$ to their destinations is built, and the node forwards $\langle \mathbf{n}, \mathcal{L}_i \rangle$ towards its destination where \mathbf{n} is the ID of this node and $\cup \mathcal{L}_i = \mathcal{L} - \mathcal{L}'$. When an intermediate node receives a reply, it puts the information in local cache as described above. In addition, each node keeps a query state:

- n : the query node ID
- $\{l_q\}$: the set of links forwarded for information and has not replied yet.

When a query $\langle n, \mathcal{L} \rangle$ comes, n and link IDs in $\mathcal{L} - \mathcal{L}'$ are stored. When a reply comes, the reply is forwarded to n and the corresponding links are removed from $\{l_q\}$. The state information is kept until $\{l_q\}$ becomes empty.

D. Dynamic Balancing of Push and Pull

The average-case optimal scope of information push depends on two key factors: the rate of information generation and the rate of request for the information [6]. Once one collects the historical information regarding these two rates in the traffic information network, one can determine the average-case optimal push scope for expected rates of information generation and request.

This average-case optimality can be further improved by capturing and responding to the fluctuations in the information generation and request rates in real-time. We propose a distributed dynamic optimization scheme called “Micro-Balancing” to achieve this goal. Instead of fixing the recipient set R at initialization or specifying it in each message, the Micro-Balancing scheme reactively changes R to minimize the total cost of information sharing.

1) *Node-link state*: Each node has a state information s with respect to a link l :

- s is **1** if the node is a recipient of link l 's information, and will rebroadcast it when receiving a new piece of information for l .
- s is **0** if s can overhear the information from one of its neighbors with state **1**.
- s is **-1** if none of its neighbors has state **1**.

For ‘‘Micro-Balancing’’, s can be changed dynamically according to the push and pull rates. To obtain the push and pull rates, a node at state **0** keeps track of the number of queries it receives and the number of information it overhears. Let N_q be the number of different queries a node receives, and N_p be the number of different ‘‘pushed’’ information a node receives, for a link l at state **0**. It is called *information-dominate* if

$$\frac{N_p}{N_q} > 2 + \sigma \quad (5)$$

and *query-dominate* if

$$\frac{N_p}{N_q} < 2 - \sigma \quad (6)$$

where σ is small value parameter for the purpose of reducing the oscillation in state-change condition boundary to reduce potential protocol overhead. The insight that leads to equations 5 and 6 is a simple one: a query and a reply needs two uses of the wireless medium and a push only need one local broadcast. An underlying assumption is that both the query and the reply can fit into one packet in the MAC layer. To make the switch between these two conditions more stable for small N_q and N_p , we compute Eq. 5 and Eq. 6 only after $N_q + N_p > \Delta$ for some constant Δ .

2) *State transition*: The rules for state transition are as follows:

- **0** \rightarrow **1**: if it becomes query-dominate,
- **1** \rightarrow **0**: if all its neighbors with state **0** are information-dominate,
- **0** \rightarrow **-1**: if none of its neighbors has state **1**
- **-1** \rightarrow **0**: if one of its neighbors has state **1**

When a node entering state **0**, both N_p and N_q are reset to 0. These state transitions are stable,

- **0** \rightarrow **1**: a node entering state **0** cannot become state **1** immediately since it takes time to accumulate N_p and N_q to be query-dominate.
- **1** \rightarrow **0**: a node entering **1** cannot become state **0** immediately, because at least one of its neighbors was in state **-1** who queried the information for the link and becomes state **0** due to the transition of this node. This node cannot transit back to **0** immediately since at least one of its neighbors who just becomes **0** is not information-dominate yet.

IV. THEORETICAL ANALYSIS

To get an insight of how the system might behave at the macro level before a simulation study, we first seek to investigate the optimal information push scope under a simplified model. Let f_i be the information generation rate of link l and c_i be the cost of push for each piece of information,

i.e., the number of transmissions needed for a specific scope. Then the total cost for information push is

$$C_{push} = f_i c_i \quad (7)$$

Assume the push scope for l is a geographical circle with radius R_i , then c_i is proportional to R_i^2 and the above equation simplifies to

$$C_{push} = \alpha f_i R_i^2 \quad (8)$$

Where α is an abstract parameter capturing ‘‘push cost per unit area’’.

Similarly, if we assume the queries for link l are generated uniformly in a circle centered at the end of l , with radius R_q and frequency per area f_q . Then the cost of information pull can be approximated as

$$C_{pull} = 2f_q(R_q^2 - R_p^2)c_q \quad (9)$$

where the factor 2 comes from the fact that query is a process involving a round trip, and c_q is an average cost for a query generated in the ring

$$c_q \sim \frac{\int_{R_p}^{R_q} (r - R_p) 2\pi r dr}{R_q^2 - R_p^2} \quad (10)$$

$$= \beta \frac{(\frac{2}{3}r^3 - R_p r^2)|_{R_p}^{R_q}}{R_q^2 - R_p^2} \quad (11)$$

where β is similar to α , capturing a corresponding notion of cost per unit distance. Therefore, C_{pull} becomes

$$C_{pull} = 2\beta f_q (\frac{2}{3}R_q^3 - R_p R_q^2 + \frac{1}{3}R_p^3) \quad (12)$$

Minimizing $C_{pull} + C_{push}$ for link l yields an optimal push and pull balance scope condition, by

$$\beta f_q R_q^2 = \alpha f_i R_i + \beta f_q R_i^2 \quad (13)$$

We can see that if f_q or f_q/f_i approaches 0, R_i should approach 0, i.e., no push is needed. On the other hand, if f_i or f_i/f_q approaches 0, $R_i \rightarrow R_q$, i.e., push as far as needed. Our simulation results in the next section verifies this theory.

V. SIMULATION RESULT

We have implemented and simulated the dynamic push and pull protocol using a commercial on-the-shelf microscopic traffic simulator Paramics [8].

A. Simulation environment and setup

Fig. 4 shows a traffic network in Paramics Simulator that we use for testing our protocol. The area of the network is 1466.6m \times 1102.3m. This network has 58 nodes and 118 links. Our protocol and event/query generations are implemented as a plug-in to the simulator. We assume each link generate some events with probability p_e at every second, and each car at the end of a link with a probability p_q queries a link that it may travel within certain time. Parameters p_e and p_q are used to control the event and query frequencies. The communication range is set to be 300m, which is a reasonable range for WiMax. The simulation data are collected



Fig. 4. A traffic network modeled in Paramics simulator.

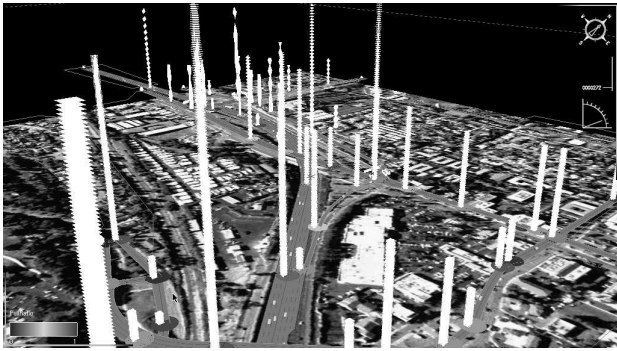


Fig. 5. Our plugin in Paramics simulator

within a time frame $[t_0, t_f]$. We record the total number of transmissions for pull and push at each node. The performance metrics is the total number of transmissions. Fig. 5 shows a snapshot of our plugin simulator, the locations of the vertical bars are the locations of the sensor nodes, and the height of bars shows the total number of transmissions at the node.

B. Simulation results

The purpose of the simulation is to verify our theoretical analysis about push and pull scope, and show the effectiveness of the dynamic push and pull protocol. Three cases are studied:

- A) Information dominate: in this case push scope should be small;
- B) Query dominate: in this case push scope should be as large as pull scope;
- C) Neither: in this case push scope should be somewhere in the middle.

For each case, we collect data for four sets of protocol settings:

- 1) push scope small,
- 2) push scope as large as pull scope,
- 3) dynamic push scope starting from small,

Cases/Settings	1)	2)	3)	4)
A)	2870	20408	2621	6808
B)	20028	1877	8619	1877
C)	20028	20408	8936	9470

TABLE I

TOTAL TRANSMISSIONS FOR THREE CASES AND FOUR PROTOCOL SETTINGS.

4) dynamic push scope starting from large.

Table I shows the result on total transmissions: We observe that for case A), push scope small is most effective, for case B), push scope large is most effective, and for case C) the ideal push scope is somewhere in the middle, and the dynamic protocol starting from any initial condition seems to be able to evolve to a lower-cost push-pull balanced configuration.

VI. CONCLUSION AND FUTURE WORK

In this paper, we introduced and studied a novel distributed vehicle traffic information system that includes a road network aware information publication protocol, a distributed query processing protocol with transient memory, and the adaptive interaction model between the two. Both theoretical model and simulation results are presented. The preliminary simulation results shows that our dynamic road-network-aware distributed push-pull balancing protocol is effective. There are many future work to be done in this direction, for instance, the cost-benefit trade-off of state switches with varying the control parameters, the effect of information caching, under a variety of traffic scenarios. Furthermore, the study of information dissemination and query on weighted directed graphs constitutes a interesting theoretical problem that worth further exploration.

REFERENCES

- [1] USDOT, "Developing traveler information systems using the national its architecture," United States Department of Transportation, 1998. [Online]. Available: <http://www.fhwa.dot.gov/tfrc/safety/pubs/its/architecture/devtravel.pdf>
- [2] ITSA, "Advanced traveler information systems: Choosing the route to traveller information systems deployment," United States Department of Transportation, 1998. [Online]. Available: <http://www.fhwa.dot.gov/tfrc/safety/pubs/its/generalits/choosette.pdf>
- [3] "Tdot smartway information system," website, August 2006. [Online]. Available: <http://www.tdot.state.tn.us/tdotsmartway/>
- [4] "Smart-traveller.com," website, 2006. [Online]. Available: <http://www.smart-traveler.info/>
- [5] S. Y. Cheng, S. C. Ergen, and P. Varaiya, "Traffic surveillance with wireless magnetic sensors," in *Proceedings of the ITS America Annual Conference*. ITS America, 2006.
- [6] X. Liu, Q. Huang, and Y. Zhang, "Combs, needles, haystacks: balancing push and pull for discovery in large-scale sensor networks," in *SensSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2004, pp. 122–133.
- [7] G. Xing, C. Lu, R. Pless, and Q. Huang, "On greedy geographic routing algorithms in sensing-covered networks," in *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*. New York, NY, USA: ACM Press, 2004, pp. 31–42.
- [8] "Paramics," website. [Online]. Available: <http://www.paramics-online.com/>