Cooperative Cache Invalidation Strategies for Internet-based Vehicular Ad Hoc Networks

Sunho Lim Chansu Yu[†] Chita R. Das[‡]
Dept. of EECS, South Dakota State University, Brookings, SD 57007, sunho.lim@sdstate.edu
[†]Dept. of ECE, Cleveland State University, Cleveland, OH 44115, c.yu91@csuohio.edu
[‡]Dept. of CSE, The Pennsylvania State University, University Park, PA 16802, das@cse.psu.edu, cdas@nsf.gov

***Abstract*—Internet-based vehicular ad hoc network** (IVANET) **is an emerging technique that combines a wired Internet and a vehicular ad hoc network** (VANET) **for providing universal information and service accessibility. A key design optimization technique in** IVANETs **is to cache the frequently accessed data items in a local storage of vehicles. Since vehicles are not critically limited by the storage/memory space and power consumption, cache replacement scheme for accommodating new data items is not an issue. Rather, a more critical design question is how to keep the cached copies valid or to invalidate them when the original data items are updated. This is particularly important in** IVANETs**, where vehicles move very fast. This paper proposes state-aware** *cooperative cache invalidation* **(CCI) scheme and its enhancement (ECCI) that take advantage of the underlying location management mechanism. Extensive performance study shows that the proposed schemes can reduce the query delay as much as 69% and increase the cache hit rate up to 57% in comparison to two existing cache invalidation techniques, called poll-each-read (PER) and extended asynchronous (EAS). Note that PER and EAS have been modified to work in** IVANETs**.**

***Index Terms*—Cooperative cache invalidation, Internet-based vehicular ad hoc network.**

I. INTRODUCTION

A vehicular ad hoc network (VANET) consists of a set of high-speed mobile vehicles equipped with communication facilities [1]. It supports inter-vehicle communications through a multi-hop message relay without the assistance of any fixed infrastructure. In order to provide a flexible connectivity, accessibility, and a rich set of services, it is imperative to consider the integration of a VANET with a wireless infrastructure, such as a wireless local area network (e.g. IEEE 802.11) and a wireless wide area network (e.g. 3G). It is envisaged that such an *Internet-based vehicular ad hoc network*, or IVANET, will prevail to become a ubiquitous communication infrastructure in the near future.

A key optimization technique in improving the communication performance of IVANETs is to cache the frequently accessed data items in a local storage. In an IVANET, it is less of a problem to determine which data items to cache because the storage space in a vehicle is not critically limited. However, a critical design issue is the *cache invalidation* scheme, since applications require data consistency with the server to avoid using any stale data. When a data item in a server is updated, it is necessary to invalidate the cached copies of this data item by broadcasting an *invalidation report* (IR). However, due to fast roaming vehicles, cache invalidation schemes developed for cellular networks and mobile ad hoc networks (MANETs) may not work well. Unlike these conventional networks, energy conservation is not an issue in IVANETs because a vehicle is supported by its own built-in battery. Rather, our concerns are query delay and cache hit ratio in the presence of mobility and frequent data updates.

The following observations characterize the IVANETs in the context of cache invalidation and support our research motivation: (i) Due to high-speed mobility, vehicles reside in a coverage area for a short period of time. For example, the connection time within a coverage area ranges from 5 to 24 seconds at city driving [2]. Therefore, when a data server broadcasts an IR, it is not straightforward which coverage area(s) to target. And, since multiple coverage areas are involved in a broadcast operation, the cost of broadcasting IRs becomes non-negligible; (ii) Since it is unlikely that adjacent vehicles have common data items in a real IVANET environment, it is wasteful to broadcast the same content of IRs to different vehicles because most of contents may not be relevant to the vehicles; and (iii) As pointed in [3], a Web proxy caching may reduce network traffics in a data server but it does not reduce the traffics in wireless links. In order to support a scalable invalidation operation with the minimized IR traffics, it is essential to coordinate with network agents of location management.

To address these issues, we propose state-aware *cooperative cache invalidation* (CCI) scheme and its enhancement, called *enhanced CCI* (ECCI), which can effectively deal with fast moving vehicles without incurring significant overhead. Our contributions are the following:

- First, we suggest a hierarchical network model for IVANETs that consists of access points (APs), gateway foreign agents (GFAs), home agents (HAs), and data server(s). This model facilitates scalable cache invalidation operations as well as efficient location management by having the GFAs to take care of micro-mobility and micro-invalidation in their respective local areas.
- Second, we propose a state-aware cooperative approach, where both a server and location management agents coordinate for cache invalidation. By maintaining a list of data items and the access history by vehicles, a server asynchronously sends an IR to an HA rather than blindly broadcasts to vehicles. Then the HA judiciously refines and distributes the IR to appropriate GFAs, which can answer the validity of the queried data item to reduce the unavoidable query delay, witnessed in most IR-based techniques.

This research was supported in part by National Science Foundation (NSF) under the Grants CNS-0831673 and CNS-0831853.

978-1-4244-4581-3/09/$25.00 ©2009 IEEE

- Third, we modify two previous cache invalidation schemes, poll-each-read (PER) and extended asynchronous (EAS), to work in IVANETs and compare them against our proposed schemes.

We conduct simulation-based performance evaluation of the four invalidation techniques along with the base case *no-cache* model to observe the impact of query interval, cache update interval, and data size on system performance, and communication overhead. According to the simulation results, the proposed schemes reduce the query delay up to 69%, increase the cache hit rate up to 57%, and incur lower communication overheads compared to two previous schemes. Overall, our study indicates that the proposed schemes provide better performance at reduced overhead and thus, are a viable approach for adopting in IVANETs.

The rest of paper is organized as follows. The prior work is analyzed and the proposed schemes are presented in Sections II and III, respectively. Section IV is devoted to performance evaluation and comparison. Finally, we conclude the paper with future directions in Section V.

II. REVISIT CACHE INVALIDATION STRATEGIES

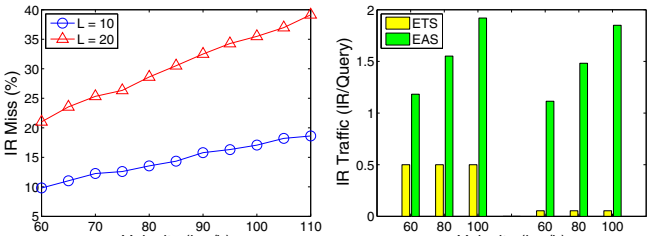
In this section, we analyze the prior cache invalidation strategies in terms of single- and multi-cell based designs, and formalize the problem in an IVANET environment.

A. Single-cell based Approach

Most of the prior IR-based schemes are variation of the timestamp (TS) scheme [4], where a base station (BS) broadcasts an IR every L seconds. Since the IR is periodically broadcasted, there is an unavoidable delay before answering a query. Depending on the length of broadcast interval, L, there is a tradeoff between the communication overhead of cache invalidation and query latency. Although an updated invalidation report (UIR) [5] is proposed to further reduce the query latency by adding updated items between IRs, query delay of half of the broadcast interval ($\frac{L}{2}$) still exists. Since the TS scheme does not consider mobility, it cannot be directly deployed in IVANETs. For example, when a vehicle generates a query and waits for an IR, it may miss the IR if it has moved to the next adjacent cell. To verify this, we experiment with a single cell environment, where 20 vehicles are randomly located and moved with a given velocity. In Fig. 1(a), as the velocity increases, more vehicles handoff before receiving the IR and thus, the IR miss ratio increases which is sensitive to L.

B. Multi-cell based Approach

To reduce the IR miss ratio due to mobility, multi-cell based designs extended from the asynchronous stateful (AS) scheme [7] are proposed, where a network component (e.g. mobile switching center (MSC) or server) located in a higher network hierarchy than a BS executes cache invalidation operations. In [6], when a mobile node handoffs, it either simply drops the cached data items or checks them with the server for validity. Since the server maintains a list of which data items are accessed by which mobile nodes, whenever a data item is



(a) Single-cell case (b) Multi-cell case

Fig. 1. Effect of velocity (L = 10 or 20 (sec) [5], [6]): (a) IR miss ratio; and (b) Average IR traffic per query (T_q = 50 (sec), and T_u = 10 (Subfig. (a)) or 100 (Subfig. (b)) (sec)).

updated, it pro-actively unicasts IRs into multiple cells, where the updated data item is cached. In VANETs, however, vehicles stay in a cell for a short period of time and travel along the large number of cells and thus, it is wasteful to examine cached data items with the server for validity upon every handoff. Also, the server's pro-active IR transmission to multiple cells is neither a scalable solution nor efficient, if the updated data item is not queried.

To verify this, we experiment with a multi-cell environment, where 50 vehicles are randomly located in five cells and move under a wrap around one-dimensional network topology. We compare two schemes extended from [4], [6], namely an extended TS (ETS) scheme and an extended AS (EAS) scheme, respectively. In ETS, a stateless server broadcasts an IR to every cell in the system whenever vehicles cache an updated data item. In the EAS scheme, however, a stateful server selectively broadcasts an IR to the cell(s). Also, upon handoff, vehicles send all the valid cached data items' ids to the server for validity and receive an invalidation check packet, which is counted as the IR traffic. In Fig. 1 (b), when the cache update interval (T_u) is 10 (sec), due to the IR broadcasts to all the cells, the ETS scheme shows the same amount of IR traffics and the impact of velocity on the IR traffic is minimized. When T_u is 100 (sec), the IR traffic reduces. However, the ETS scheme is not scalable in terms of the cache update rate and network size. In the EAS scheme, as the velocity increases, it shows very high IR traffic regardless of cache update interval because of additional invalidation check packets upon every handoff (Here, the mean query interval (T_q) is set to 50 (sec).).

C. Summary

Although the ETS scheme reduces the impact of mobility, there still exists a non-negligible query delay and the stateless server blindly broadcasts an IR to entire cells in the system, resulting in a scalability issue. On the other hand, the EAS scheme has no query delay when a queried data item is cached and marked as valid, but it incurs heavy IR traffic mainly due to mobility. Thus, reduction of the mobility impact and IR traffic are conflicting requirements, and optimization of both is extremely complex.

III. THE PROPOSED CACHE INVALIDATION STRATEGY

In this section, first we briefly introduce a system model and then present our cache invalidation techniques.

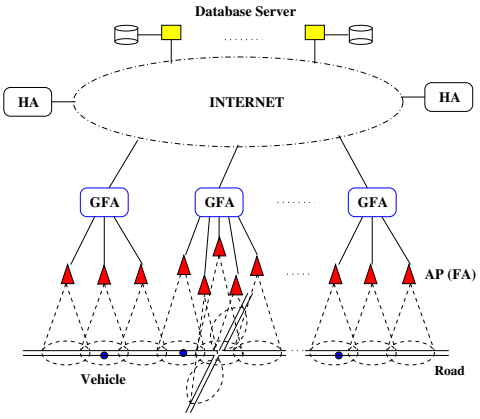


Fig. 2. A system model of IVANET.

A. System Model

As illustrated in Fig. 2, a hierarchical network model for IVANET is motivated by a cellular network, and it consists of access points (APs), gateway foreign agents (GFAs), home agents (HAs), and servers of data item source. The AP plays an additional role as a foreign agent (FA). First, vehicles are equipped with communication facilities such as an IEEE 802.11-based dedicated short range communication (DSRC) transceiver. Thus, they can either communicate with other vehicles or connect to the Internet flexibly. Unlike cellular and MANET environments, where nodes move without restrictions, vehicles are bounded to the underlying fixed roads with speed limits and traffic lights in VANETs. The vehicles generate a read-only query for a data item, which is the unit of an update or a query operation and stored in a database attached to a database server. The database is only updated by the server.

Second, in order to support global wireless access of an Internet, a mobile IP [8] is deployed, in which vehicles are able to access the Internet and maintain their on-going communications, while they move at a high-speed. To avoid frequent location register/update operations for fast roaming vehicles, we use an IP micro-mobility support, in which vehicles report the HA only when a major change occurs such as changes to a regional network. In this paper, a 2-way handshake [9] is used to implement location updates by exchanging either a home or regional request/reply packet between a vehicle and a GFA (or HA) through an AP (or GFA).

B. The State-aware Cooperative Approach

In this paper, we propose a *state-aware cooperative approach*, where a server and network agents of location management coordinate together for cache invalidation operations. The server and network agents maintain a list of which data item is accessed by which vehicle and vehicles' current locations, respectively. Unlike most of the prior stateful designs, the server does not keep track of a vehicle's current location for IR broadcast. Thus, whenever a data item is updated, it sends an IR to an HA rather than blindly broadcasting the IR to multiple cells, where the updated data item is cached in a vehicle. Then the HA judiciously refines and re-distributes the IR to appropriate GFA(s), where they can answer the validity of a queried data item, requested from a vehicle. Here, GFAs do not pro-actively send the IR to the individual vehicles, but reply a vehicle's validity request by a on-demand-basis.

The rationale behind this approach is the following: (i) Since a vehicle's location is implicitly maintained by both a GFA and an HA, the server can avoid keeping track of the vehicles' current location, which is in fact a duplicated operation with the location management; (ii) Also, since a vehicle has a low probability of finding common data items in adjacent vehicles, broadcasting the same IR to different vehicles is inefficient in IVANETs; and (iii) In addition, since the query rate of vehicles is independent of the cache update rate of the server, whenever a data item is updated, a blind IR broadcast incurs heavy IR traffic. The proposed cache invalidation mechanism avoids these issues.

C. Proposed Cache Invalidation Technique

In this subsection, we describe the detail operation of the cooperative cache invalidation (CCI) scheme. First, a server maintains a list of which data item (d_x, $x \in D$) is accessed by which vehicle in a registry (r_s), $[id(d_x), [vid(v_{x,y}), t_{x,y}]]$, where $v_{x,y} = \{v_y | \text{request}(d_x) \wedge (y \in N)\}$, vid is a vehicle's id, and $t_{x,y}$ is the access time of d_x by v_y. Here, D and N are the number of data items and vehicles, respectively. Whenever a data item is updated, the server generates an IR ($\mathcal{R}_s$), $< id(d_x), vid(v_{x,y}), t_{cur} >$, and it is not directly broadcasted to the vehicles but sent to an HA. Here, t_{cur} is the current timestamp. Upon receiving the $\mathcal{R}_s$, since the HA maintains location information about which vehicle is currently roaming under which GFA (g_i) in a registry (r_h), $[vid(v_y), g_i]$, it compares the v_y with $\mathcal{R}_s$ and r_h. Then the HA creates a new IR ($\mathcal{R}_h$), $< id(d_x), t_{cur} >$, based on the matched g_i and sends it to the appropriate GFA. Depending on the number of matched GFAs, more than one $\mathcal{R}_h$ can be created. Also, each $\mathcal{R}_h$ may have a different size that is less than or equal to the original $\mathcal{R}_s$ received from the server. Upon receiving the $\mathcal{R}_h$, the GFA updates its registry and thus, it has the most recent updated IR with the server.

Second, when a vehicle (v_y) initially connects to a server, it sends all the ids of cached data items (d_k^c) and their timestamps, $[vid(v_y), id(d_k^c), t_k]$, and then the server updates its registry. Here, k is $1 \leq k \leq c$ and c is the total number of cache slots. Whenever the server receives a request packet for validating or accessing a data item, it also updates its registry. Also, when a vehicle handoffs to the coverage area which is under a different GFA, then the HA updates its registry and forwards the $\mathcal{R}_h$ to the new GFA when a cached data item is updated in the vehicle. The $\mathcal{R}_h$ may be forwarded to the appropriate APs to further reduce the query latency, but it is an overhead for the GFA, if a vehicle frequently moves among the coverage areas.

According to the proposed cache invalidation mechanism, let us examine the detail operation of a query request. First, when a vehicle (v_y) generates a query for a data item (d_x) which is not cached, it sends a request packet for accessing the data item to the server. The packet contains the vehicle's care-of address (CoA), an id of the requested data item, and a single bit flag (f) representing whether the queried

data item is cached or not, $[CoA_y, id(d_x), f_x]$, and it is sent to the nearest AP. Here, f_x is set to 0. As the packet is forwarded to the server through the AP and GFA, CoAs of the AP and GFA are appended in the packet header, $[CoA_{GFA}, [CoA_{AP}, [CoA_y, id(d_x), f_x]]]$, to keep the route information. Upon receiving the request packet, the server attaches the queried data item to the ack packet with the route information. The ack packet is replied back to the AP via the GFA. Then the AP unicasts the packet and thus, the corresponding vehicle can receive the data item.

Second, when a query is generated which can be answered by a cached copy of data item, the vehicle sends a request packet to the GFA for checking validity through the nearest AP. The GFA compares the id of queried data item with the $\mathcal{R}_h$ and replies an ack packet, if the cached copy is valid. If the cached copy is not valid, then the GFA forwards the packet to the server. When the server receives the request packet, it verifies the status of the queried data item and replies an ack packet with the route information, if it is a valid copy. If not, it uses the same procedure as the first case to supply the data item.

In summary, based on the state-aware cooperative approach, the server and network agents of the underlying location management coordinate the cache invalidation operations, where the server does not blindly broadcast an IR but sends it to the HA. Then the HA judiciously refines and re-distributes the IR to the GFAs, where the queried data items can be validated.

D. An Enhancement

In this enhanced cooperative cache invalidation (ECCI) scheme, the GFAs cache data items in their local storage for future query requests from vehicles. Since an updated data item attached to an ack packet is replied back to vehicles from the server, a GFA intercepts the packet and caches the updated data item. Based on the IR received from the server through HA, GFAs also can maintain a set of valid data items and reply ack packets to vehicles directly, if the queried data item is cached, without forwarding the query request to the server.

IV. Performance Evaluation

A. Simulation Testbed

We develop a customized discrete-event simulator using CSIM19 [10] to conduct our experiments. To examine the proposed idea, we use a wrap around one-dimensional network topology, where 25 APs[1] covers the area. We assume that each AP is located in the center of a coverage area with 2 Mbps bandwidth. Five GFAs are deployed in the area and each GFA consists of five APs. A set of vehicles is randomly located in the area, travels with a given velocity, and communicates with the APs. The communication delay is dependent on the distance among wired network agents represented by the number of hops. Based on [9], the number of hops between an AP and a GFA, a GFA and an HA, a GFA and the server, and the server and an HA are set to 5, 10, 25, and 25, respectively. We assume that the bandwidth of the wired network is 100 Mbps.

TABLE I
Simulation parameters

Parameter	Value
Network size (km)	18.85
Diameter of AP coverage (m)	750 [11]
Velocity (km/h)	60
Number of vehicle	250
Database size (items)	1,000
Data item size (KByte)	1 - 500
Hot data items	50
Cold data items	Remainder of DB
Hot data item access prob.	0.8

Both query and update inter arrival times follow the exponential distribution. The entire data items stored in the database is classified into two subsets, hot and cold data items. We assume that 80% of query requests are for hot data items, and an update request is uniformly distributed within the hot and cold subsets. The Zipf distribution model [12] is often used to model a skewed access pattern, where θ is the access skewness coefficient. The access probability of the i^{th} data item is represented as $\frac{\Omega}{i^\theta}$, where $\Omega = (\sum_{j=1}^{n} \frac{1}{j^\theta})^{-1}$, and $0 \leq \theta \leq 1$. Setting θ = 0 corresponds to the Uniform distribution. Here, we set θ to 0.8 based on the real Web traces [13].

Each vehicle caches 5% of the data items in the database. When a cache is full, we use the least recently used (LRU) cache replacement policy. We do not deploy an advanced cache admission or replacement policy such as [14] to clearly see the effect of the proposed scheme on the performance. The simulation parameters are summarized in Tab. I.

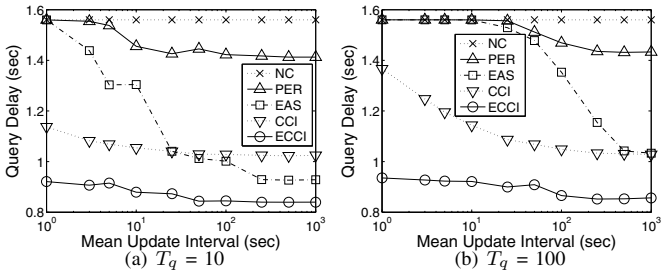
B. Performance Comparison

To compare the performance of proposed schemes, we modified two prior cache invalidation techniques to work in IVANET: a poll-each-read (PER) scheme, and an extended asynchronous (EAS) scheme. The PER[2] scheme [15] is an on-demand approach extended from [16], where a query is forwarded to the server for either validating a cached data item or receiving the queried data item. In the EAS[3] scheme, as we mentioned in section II, a stateful server maintains a list of ids of the cached data items in vehicles, and broadcasts an IR to inform the vehicles of any cache update. Thus, a queried data item, which is cached, is used to answer the query directly without validating with the server. When a vehicle handoffs, it sends the entire valid cached data item ids to the server for validity check and receives an invalidation check packet from the server. For comparison, we add the proposed cooperative stateful approach in these schemes, where the impact of mobility on the performance is minimized. In addition, we

[1]The AP (e.g. an infostation or a message relay box) is located along the road and acts as a gateway to an infrastructure network. We envision that drivers should be able to access not only roadside Wi-Fi stations but also high-speed wide area 3G cellular networks and thus, the diameter of coverage area will become wider than that of we have assumed based on [11] in this paper.

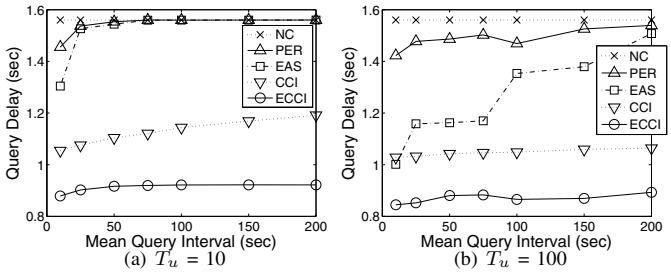
[2]A server-based poll-each-read (SB-PER) scheme [15] is proposed to further improve the performance of PER under the assumption of available global access and update information in a server. However, obtaining the global update information is practically infeasible and thus, we do not consider the SB-PER scheme in this paper.

[3]Both asynchronous (AS) [6] and call-back (CB) [17] schemes are similar in the sense that a stateful server broadcasts an IR to the mobile nodes for cache updates. Since the CB scheme is designed for a single-cell environment, we extend the AS scheme for comparison in this paper.



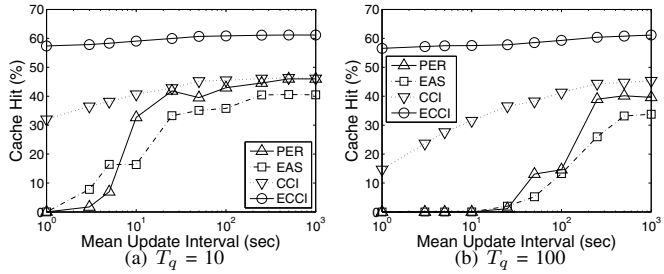
(a) T_q = 10 (b) T_q = 100

Fig. 3. Query delay as a function of mean update interval.



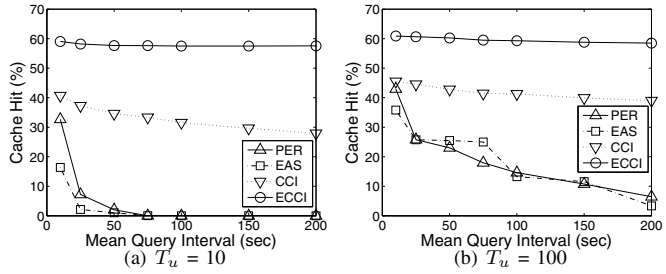
(a) T_u = 10 (b) T_u = 100

Fig. 5. Query delay as a function of mean query interval.



(a) T_q = 10 (b) T_q = 100

Fig. 4. Cache hit as a function of mean update interval.



(a) T_u = 10 (b) T_u = 100

Fig. 6. Cache hit as a function of mean query interval.

include the base case, no cache (NC), where a query request is forwarded to the server always.

C. Simulation Results

In this section, we evaluate the impact of update interval (T_u), query interval (T_q), and size of data items, and examine the communication overheads of the cache invalidation strategies.

1) Impact of Update Interval: First, we evaluate the query delay and cache hit rate of cache invalidation strategies as a function of update interval. In Fig. 3(a), as the update interval increases, the query delay decreases. The NC scheme is not affected by the update and query intervals because every query is forwarded to the server for access. In Fig. 3(b), when the query interval is high, the overall query delays increase because more cached data items become invalid before they are queried. Both PER and EAS schemes show high query delays closer to the NC scheme in the low update intervals, but the EAS scheme exhibits lower query delay with high update intervals, because more valid cached data items are found without additional validation delay with the server. Both our CCI and ECCI schemes show better performance than other two schemes. Especially the ECCI scheme shows the lowest query delay for the entire update intervals due to additional caching in the GFAs.

In Fig. 4(a), as the update interval increases, the cache hit rate increases. Both CCI and ECCI schemes show higher cache hit rates than other two schemes because a query can be validated or answered in the GFA before it is forwarded to the server. The detail analysis of cache hit is presented in Fig. 7(a) later. In Fig. 4(b), when the query interval is high, the overall cache hits decrease. However, the ECCI scheme is not affected much by both query and update intervals and shows the highest cache hit rate for the entire update intervals.

2) Impact of Query Interval: Second, we examine the query delay and cache hit of the cache invalidation strategies as a function of query interval. In Fig. 5(a), when the update interval is low, both PER and EAS schemes show almost similar performance to the NC scheme except for the low query intervals. This is because, due to frequent update of data items, more queried cached data items become invalid and thus, more queried requests are forwarded to the server. In Fig. 5(b), when the update interval is high, the overall query delays decrease. The EAS scheme shows lower query delay than the PER scheme because a valid cached data item can directly be used for answering a query, but higher query delay than that of both CCI and ECCI schemes because a query is forwarded upto the server when the queried cached data item is either invalid or unavailable.

In Fig. 6(a), when the update interval is low, both PER and EAS schemes show almost no cache hit except for the low query intervals due to frequent update of data items. In Fig. 6(b), when the update interval is high, both PER and EAS schemes still suffer from the low cache hit resulting in the high query delay. Regardless of query and update intervals, both the CCI and ECCI schemes achieve higher cache hits than the other two schemes.

3) Impact of Cooperative Stateful Approach: Third, we analyze the proposed schemes in terms of the local cache hit and a remote cache hit. The local cache hit is the ratio that a queried data item is locally cached and validated by the server, while the remote cache hit is the ratio that a queried data item is validated or found from the GFAs. In the ECCI scheme, the remote cache hit can be enhanced when a queried data item is cached in GFAs. In Fig. 7(a), when the pair of query and update interval is 10/100 (sec), both CCI and ECCI schemes show higher cache hits than other two schemes. The CCI scheme achieves the highest remote cache hit because many queries are validated in the GFAs before they are forwarded to the server. However, the ECCI scheme shows higher local cache hit than other three schemes because many queried data items found in the GFA brought to the query request

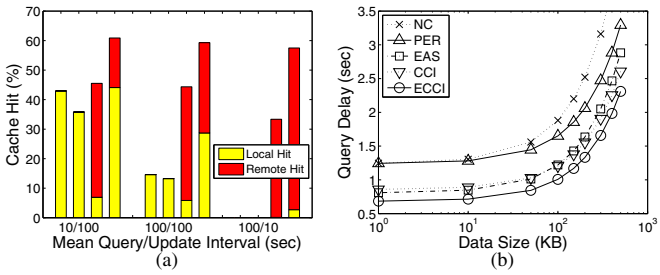


Fig. 7. The left figure shows the cache hit against the pair of mean query and update intervals, where the PER, EAS, CCI, and ECCI schemes are plotted from left to right. The right figure shows the query delay against data size (T_q = 10 and T_u = 50).

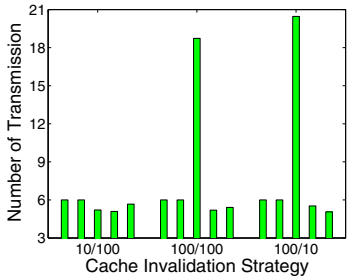


Fig. 8. Average number of transmissions per query is shown against the pair of mean query and update intervals. The NC, PER, EAS, CCI, and ECCI schemes are plotted from left to right.

senders for caching. Then these cached data items are used for answering the next query requests, resulting better local cache hits. In case of 100/10 (sec), due to frequent update of data items, both PER and EAS schemes show no cache hit, while both CCI and ECCI schemes heavily rely on the remote cache hit.

4) Impact of Data Size: Next, we measured the query delay as a function of data size. In Fig. 7(b), the ECCI scheme outperforms the others and achieves the lowest query delay for all data sizes. The performance gap between the NC and other schemes is larger as the data size increases. Due to additional communication with the server on every query request for validating or accessing the data items, both NC and PER schemes show higher query delays than the other three schemes.

5) Communication Overhead: Finally, we evaluate the communication overhead of the cache invalidation strategies in terms of the average number of packet transmissions only from the network components including the query request, reply, and IR packet transmissions. In Fig. 8, both NC and PER schemes require six packet transmissions from a vehicle, AP, GFA, and server for answering a query. Although additional IR transmission from the server and HA is required in both CCI and ECCI schemes, they achieve the less number of packet transmissions than that of NC and PER schemes because some of queries can be validated in GFAs before they are further forwarded to the server. The EAS scheme may reduce the query delay, but it incurs maximum communication overhead, especially in low update intervals (e.g. 100/10 (sec)) mainly due to excessive IR packet transmissions.

In summary, the proposed cooperative cache invalidation schemes provide better performance and less overhead and thus, are a viable approach for implementing IVANETs.

V. Concluding Remarks and Future Work

In this paper, we investigate the cache invalidation issue in an IVANET, where minimizing the impact of high-speed mobility and designing a scalable algorithm are primary concerns. We proposed a hierarchical network model and a state-aware cooperative cache invalidation scheme including its enhancement, in which both the server and network agents of location management coordinate the cache invalidation operation. We compared the proposed schemes with two prior cache invalidation schemes through extensive simulation, and observed that the proposed CCI and ECCI schemes provide better performance than others with respect to the query delay, cache hit rate, and communication overhead.

For future work, we plan to investigate an adaptive cache consistency mechanism in IVANETs. In this paper, we implicitly assume a strong consistency model, in which a query is answered by the latest updated data item from either a local cache or the server. We plan to relax this assumption to support growing diversity in applications and users' demands requiring a certain consistency level with the server.

References

[1] *Dedicated Short Range Communications (DSRC) Home*, http://www.leearmstrong.com/DSRC/DSRCHomeset.htm.

[2] V. Bychkovsky, B. H. A. Miu, H. Balakrishnan, and S. Madden, "A Measurement Study of Vehicular Internet Access Using in Wi-Fi Networks," in *Proc. ACM MOBICOM*, 2006, pp. 50–61.

[3] H. Chen and Y. Xiao, "Cache Access and Replacement for Future Wireless Internet," *IEEE Communications Magazine*, pp. 113–123, 2006.

[4] D. Barbara and T. Imielinksi, "Sleepers and Workaholics: Caching Strategies for Mobile Environments," in *Proc. ACM SIGMOD*, 1994, pp. 1–12.

[5] G. Cao, "A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments," in *Proc. ACM MOBICOM*, 2000, pp. 200–209.

[6] Z. Wang, M. Kumar, S. K. Das, and H. Shen, "Investigation of Cache Management Strategies for Multi-cell Environments," in *Proc. 4th International Conference on Mobile Data Management (MDM)*, 2003, pp. 29–44.

[7] S. Khurana, A. Kahol, S. Gupta, and P. Srimani, "A Strategy to Manage Cache Consistency in a Distributed Mobile Wireless Environment," in *Proc. IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2000, pp. 530–537.

[8] C. E. Perkins, *IP Mobility Support*, Request for Comments (RFC) 2002-2006.

[9] J. Xie and I. F. Akyildiz, "A Distributed Dynamic Regional Location Management Scheme for Mobile IP," in *Proc. IEEE INFOCOM*, 2002, pp. 1069–1078.

[10] *The CSIM User Guides*, http://www.mesquite.com/documentation/index.htm.

[11] S. Yoon, H. Q. Ngo, and C. Qiao, "On "Shooting" a Moving Vehicle with Data Flows," in *Proc. Mobile Networking for Vehicular Environments (MOVE-07)*, 2007, pp. 49–54.

[12] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Cambridge, MA, 1949.

[13] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications," in *Proc. IEEE INFOCOM*, 1999, pp. 126–134.

[14] S. Lim, W. Lee, G. Cao, and C. R. Das, "A Novel Caching Scheme for Improving Internet-based Mobile Ad Hoc Networks Performance," *Ad Hoc Networks Journal*, vol. 4, no. 2, pp. 225–239, 2006.

[15] H. Chen, Y. Xiao, and X. Shen, "Update-Based Cache Access and Replacement in Wireless Data Access," *IEEE Trans. on Mobile Computing*, vol. 5, no. 12, pp. 1734–1748, 2006.

[16] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, and M. West, "Scale and Performance in a Distributed File System," *ACM Transactions on Computer Systems*, vol. 6, no. 1, pp. 51–81, 1988.

[17] Y. Xiao and H. Chen, "An Adaptive Callback Cache Access for Wireless Internet," in *Proc. IEEE GLOBECOM*, 2005, pp. 1049–1053.