Data Consistency for Cooperative Caching in Mobile Environments

Jiannong Cao, Hong Kong Polytechnic University
Yang Zhang and Guohong Cao, Pennsylvania State University
Li Xie, Nanjing University

A 3D framework provides a basis for designing, analyzing, and evaluating strategies to address data consistency issues in mobile wireless environments. A proposed relay-peer-based cache consistency protocol offers a generic and flexible method for carrying out cache invalidation.

The trend toward wireless communications and advances in mobile technologies are increasing consumer demand for ubiquitous access to Internet-based information and services. However, due to battery power limitations, users often must disconnect mobile devices from the network to conserve energy. Moreover, wireless links have lower capacity than wired links and wireless channels are less stable, resulting in higher network congestion and packet loss. These challenges make mobile communication unreliable, emphasizing the need for efficient information-access mechanisms.[1-3]

Cooperative caching improves system performance[4,5] because it allows sharing and coordination of cached data among multiple mobile users in the network. By cooperatively caching frequently accessed information, mobile devices do not always have to send requests to the data source. In addition to reducing query latency, this technique can lower mobile host communication overhead and energy consumption. However, the unreliable communication and the user's mobility make it difficult to maintain cache consistency.

Mobile wireless environments can be broadly classified as either infrastructure based or ad hoc based. In the former scheme, shown in Figure 1a, a fixed network device such as a mobile support station forwards messages that mobile hosts send or receive. The MSS is similar to the server in a traditional client-server distributed system in that all source data is deployed on it. Other mobile hosts retrieve data from the MSS and can cache a replica by themselves. In contrast, ad hoc networks, like that shown in Figure 1b, do not store data on the MSS but use it only as the access point to the Internet. Ad hoc networks disperse all data items for searching and querying across the mobile hosts. Thus, the cache invalidation strategies employed in a single-hop wireless mobile network are not suitable for a multihop mobile ad hoc network.

As the "Cache Invalidation Techniques" sidebar describes, researchers have proposed a wide range of strategies for maintaining cache consistency in recent years. However, each strategy has its own design goals and application scenarios. No uniform or structured methods exist for current work, making it difficult to evaluate their relative effectiveness and performance.

In response to this problem, we have developed a 3D model that captures the main features of cache consistency schemes and provides a basis for evaluating existing strategies as well as designing new ones. Based on this model, we propose a hybrid and generic strategy: *relay-peer-based cache consistency*. Because RPCC uses relay peers between the source hosts and cache hosts to forward updated information, it can divide cache invalidation into two asynchronous procedures. Moreover,

Published by the IEEE Computer Society

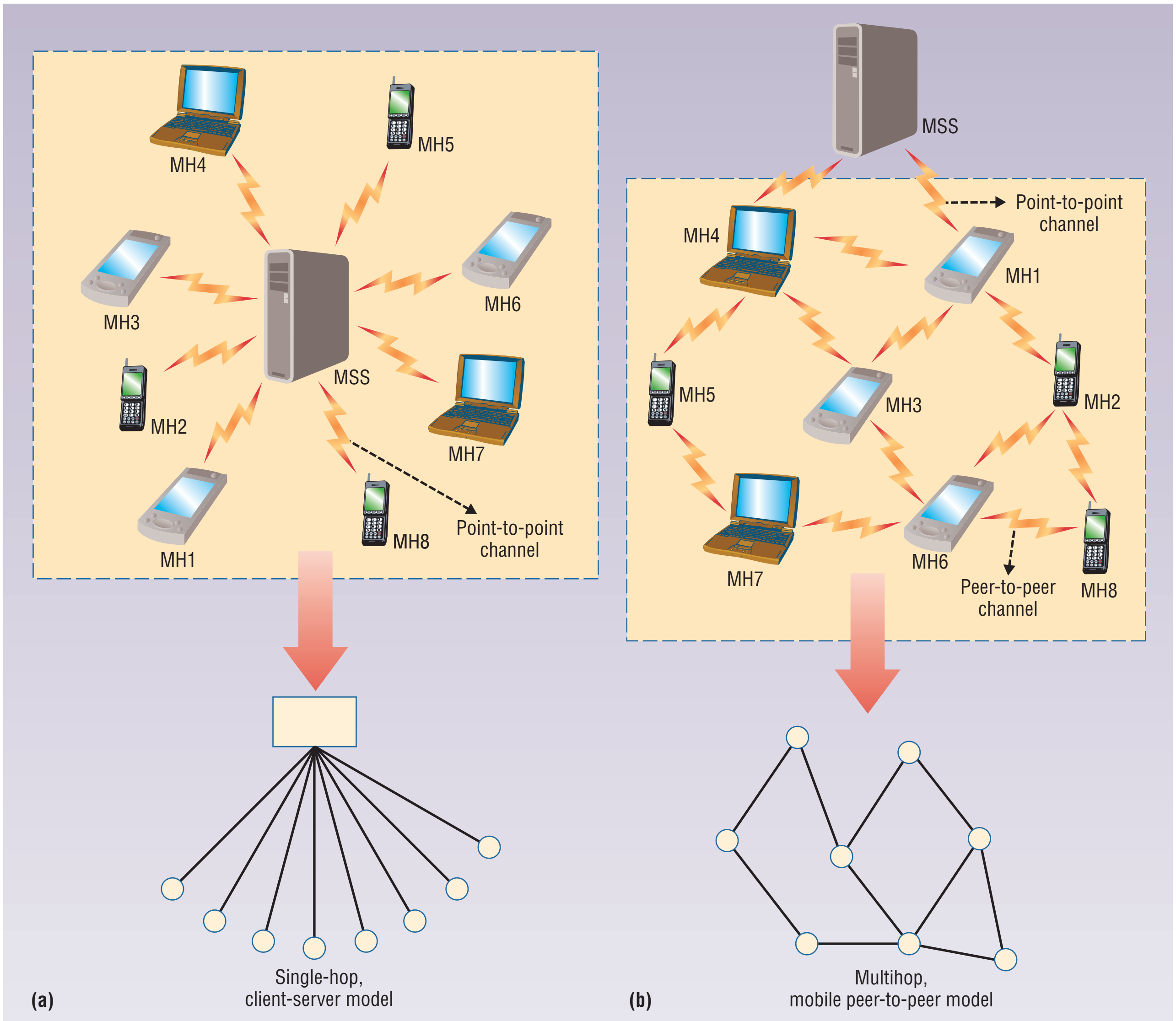

Figure 1. Wireless mobile environments. (a) In infrastructure-based single-hop networks, the mobile support station forwards messages that mobile hosts send or receive. (b) Ad hoc multihop networks use the mobile support station only as the access point to the network.

RPCC caters to different application requirements by providing a flexible and convenient way to set the consistency level and distance between the source data hosts and relay peers.

DESIGN ASPECTS

A strategy for maintaining consistency of cooperatively cached data in a mobile wireless network can focus on three design aspects: *consistency level*, *consistency control*, and *cache status maintenance*.

Consistency level

There are three levels of cache consistency: strong, delta, and weak. In each case, let S^t_{Di} denote the version number of data item D_i at the source node, and let $C^t_{Di,Mj}$ denote the time stamp of D_i at the cache node M_j at time t. The version number is set to zero when the data item is created and increases incrementally upon each subsequent update.

If a version of D_i at the cache node is always up to date with the source data when a query request arrives, it satisfies *strong consistency* (SC): $\forall t, \forall j, C^t_{Di,Mj} = S^t_{Di}$. If any read of D_i is never out of date by more than Δ time with the source data at the source node, it satisfies *delta consistency* (DC): $\forall t, \forall j, \exists \tau, 0 \le \tau \le \Delta$, stale time $C^t_{Di,Mj} = S^{t-\tau}_{Di}$. If a read of D_i does not necessarily reflect the most up-to-date version of the source data but some correct previous value, it satisfies *weak consistency* (WC): $\forall t, \forall j, \exists \tau$, stale time $C^t_{Di,Mj} = S^{t-\tau}_{Di}$.

Consistency control

Both the source and the cache nodes can initiate consistency checking. In the former approach, invalidation

Cache Invalidation Techniques

Cache invalidation techniques have been widely used in distributed systems to maintain data consistency among caches. Researchers have adapted these techniques to both single-hop and multihop wireless networks, but the proposed solutions cannot suitably deal with resource constraints, frequent disconnections, or high client mobility.

Single-hop strategies

Daniel Barbará and Tomasz Imielinski[1] proposed three algorithms—*time stamps, amnesic terminals,* and *signature*—in which the mobile support station broadcasts an invalidation report indicating the updated data items. Rather than querying the MSS directly regarding validation of the cached copies, the mobile hosts can listen to these IRs over the wireless channel. Due to its limited size, however, an IR can only record the updated information in intervals, making it useless to any mobile host that has been disconnected longer.

Kun-Lung Wu, Philip S. Yu, and Ming-Syan Chen[2] as well as Jin Jing and colleagues[3] made some modifications to the traditional IR-based strategies to handle the long disconnection problem. However, the querying mobile host must listen to the next IR to check the invalidation status, making the average waiting time longer than half of the IR interval.

An invalidation technique can reduce the query latency by inserting several updated invalidation reports between two successive IRs.[4] If the data updates frequently, however, the UIR will become very large, thus aggravating network traffic.

Anurag Kahol and coauthors[5] proposed an *asynchronous stateful* strategy to maintain cache consistency. In AS, the MSS only broadcasts the data update to the related cache hosts and can avoid unnecessary IRs. However, the MSS must record the state of all caches, which consumes too many source node resources.

In the *scalable asynchronous cache consistency scheme* developed by Zhijun Wang and colleagues,[6] the MSS only keeps minimum state information instead of all mobile hosts' information as in AS. This can improve scalability and performance, but stale data can exist in the caches.

Multihop strategies

Jian Lan and coauthors introduced the consistency maintenance technique used in a Gnutella-like peer-to-peer file-sharing network.[7] They described three strategies for keeping the cache consistency: *simple push*, *simple pull*, and *push with adaptive pull*.

Two cache invalidation mechanisms proposed by Sunho Lim and colleagues,[8] *aggregate cache based on demand* and *modified time stamp*, can improve cache invalidation performance with the help of the Global Positioning System, but the expense of GPS devices restricts its popularization.

References

1. D. Barbará and T. Imielinski, "Sleepers and Workaholics: Caching Strategies for Mobile Environments," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* ACM Press, 1994, pp. 1-12.
2. K-L. Wu, P.S.Yu, and M-S. Chen, "Energy-Efficient Caching for Wireless Mobile Computing," *Proc. 12th Int'l Conf. Data Eng.,* ACM Press, 1996, pp. 336-343.
3. J. Jing et al., "Bit-Sequences: An Adaptive Cache Invalidation Method in Mobile Client/Server Environments," *Mobile Networks and Applications,* vol. 2, no. 2, 1997, pp. 115-127.
4. G. Cao, "A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments," *IEEE Trans. Knowledge and Data Eng.,* vol. 15, no. 5, 2003, pp. 1251-1265.
5. A. Kahol et al., "A Strategy to Manage Cache Consistency in a Distributed Mobile Wireless Environment," *IEEE Trans. Parallel and Distributed Systems,* vol. 12, no. 7, 2001, pp. 686-700.
6. Z. Wang et al., "A Scalable Asynchronous Cache Consistency Scheme (SACCS) for Mobile Environments," *IEEE Trans. Parallel and Distributed Systems,* vol. 15, no. 11, 2004, pp. 983-995.
7. J. Lan et al., "Consistency Maintenance in Peer-to-Peer File-Sharing Networks," *Proc. 3rd IEEE Workshop Internet Applications,* IEEE Press, 2003, pp. 90-94.
8. S. Lim et al., "Performance Comparison of Cache Invalidation Strategies for Internet-Based Mobile Ad Hoc Networks," *Proc. IEEE Int'l Conf. Mobile Ad-Hoc and Sensor Systems,* IEEE Press, 2004, pp. 104-113.

messages are broadcast to all the cache nodes indicating the update status of data items; in the latter approach, the cache node polls the owner to determine whether its cached data item is stale or not. The source-initiated approach puts the burden of consistency maintenance on the source node—it is a kind of *push* (PS) operation in that the source node pushes the update message to other cache nodes. In contrast, the cache-initiated approach is a kind of *pull* (PL) operation that needs individual cache nodes to pull the update messages from the source data.

Cache status maintenance

The source node may or may not record the status of its cache nodes. In the *stateful* (SF) approach, the source keeps the information about which mobile hosts cache which data items. When the source node wants to communicate with these cache nodes, it can send unicast or multicast messages instead of broadcasting them. This approach is more efficient and saves bandwidth, but because the source node must store the cache states for all the cache nodes, it is not scalable to a large population of mobile users.

The *stateless* (SL) approach does not require the source node to be aware of the cache nodes' state. Instead, it keeps track of the update history (of a reasonable length) and periodically floods this information in the form of an invalidation message, creating considerable network communication overhead.

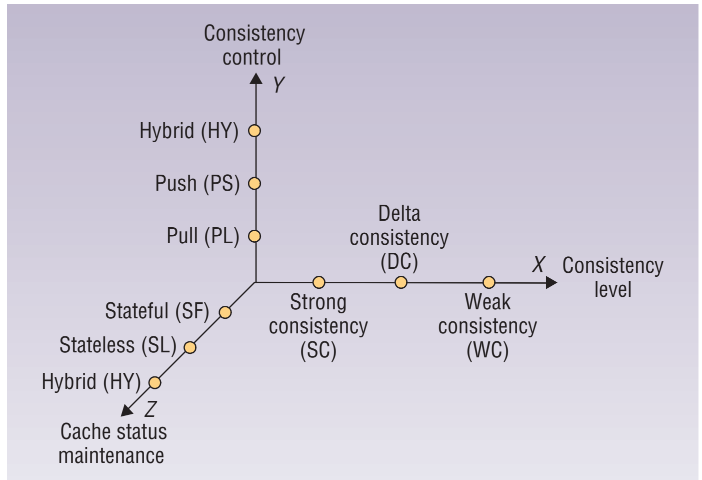

Figure 2. 3D design framework. Each axis represents one design aspect and contains a range of properties. Combining parameters from all three dimensions yields a taxonomy of caching consistency maintenance strategies.

3D DESIGN FRAMEWORK

Figure 2 shows a proposed 3D framework that represents each design aspect as one orthogonal dimension. Because the three dimensions are independent of one another, a wide variety of cache consistency maintenance schemes can be created by combining properties from each.

An XX-YY-ZZ string expresses a strategy in which XX represents consistency level (SC, DC, or WC), YY stands for consistency control (PS, PL, or HY—a hybrid of PS and PL), and ZZ symbolizes cache status maintenance (SF, SL, or HY). A strategy's overall configuration has a special value for each parameter. Some XX-YY-ZZ combinations are meaningful in a mobile environment; in fact, all the consistency maintenance strategies in the literature have corresponding XX-YY-ZZ tuples.

SC-PL-SL is identical to the *poll each read* strategy[6] in a traditional distributed system. Before accessing a cached data item, the user must query the source node for the status of its cached data. This is the simplest consistency strategy but has a poor query performance and imposes significant load on the source node. *Aggregate cache based on demand* for multihop mobile environments is also a kind of SC-PL-SL scheme.

Poll with time-out period[7] is a kind of DC-PL-SL strategy that assumes cached data remains valid for at least a time-out period of t after a cache node validates the data. When $t = 0$, this strategy is equivalent to polling each read. In DC-PL-SL, a long time-out increases the likelihood that caches will supply stale data to users, but data will be less than t stale.

In SC-PS-SF, the source node keeps track of which nodes are caching which data items. Before modifying its source data, the source node will notify all of its cache nodes. *Callback*[8] and *lease-based*[9] strategies are of this type.

The *-PS-SL strategies (in which * represents those situations with nonfixed attributes and can be any combination of properties along the dimension) have been widely employed and include most schemes based on *invalidation reports*. Each cache node can validate the cached data only when it receives the IR from the source node.

The *-PS-SF strategy is similar to *-PS-SL except that the source node takes a stateful approach. *Asynchronous stateful* is a kind of SC-PL-SF strategy. The source—for example, the MSS—records the data information of each cache node and only broadcasts the update to the related cache nodes, which helps to reduce network traffic. The *scalable asynchronous cache consistency scheme* is also a type of SC-PS-SL.

Push-based strategies are more suitable to a stable network. They can provide good consistency guarantees for users who are always online and reachable from the source. However, such strategies have low query latency and cannot solve the disconnection problem. If the cache nodes are disconnected from the network, they cannot receive the invalidation messages and will share the stale data upon reconnection.

Pull-based strategies are suitable for dynamic networks, but message flooding induces high communication overhead. The cache node's on-demand polling will consume much battery power while the mobile device's energy is quite limited. The advantage of the SF approach is that the source node avoids unnecessary broadcast flooding to the network, but to maintain all the cache information of mobile hosts, the source node must maintain a large and complicated database. The SL approach is easy to implement and manage, but more unnecessary broadcast messages will flood the networks.

RELAY-PEER-BASED CACHING CONSISTENCY

RPCC is a novel HY-HY-* strategy that combines the advantages of both push-based and pull-based approaches while avoiding their weaknesses.[10] It effec-

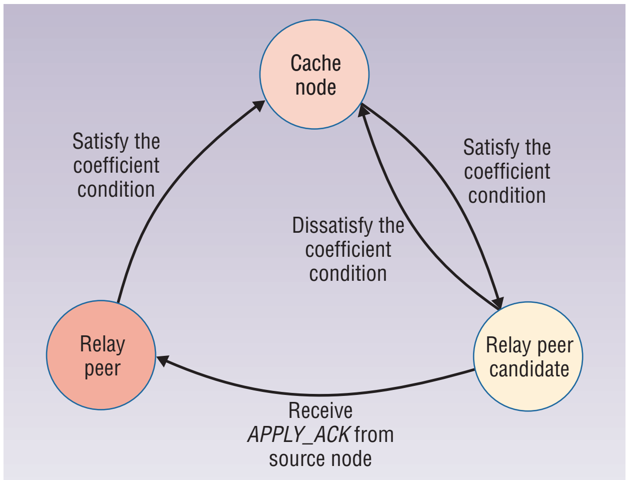

Figure 3. State transition from cache node to relay peer. A cache node becomes a relay-peer candidate if its coefficients satisfy a criterion.

Table 1. GloMoSim simulation parameters.

Parameter description	Defined value
Communication range	250 meters
Simulation time	5 hours
Update interval	2 minutes
Query interval	20 seconds
TTL in simple pull-push	8 hops
TTL in RPCC	3 hops
TTN of source node	2 minutes
TTR of relay peer	1.5 minutes
TTP of cache node	4 minutes

TTL = time to live; TTN = time to notify; TTR = time to refresh; TTP = time to poll

tively maintains cache consistency in mobile wireless environments, especially ad-hoc-based ones, by selecting highly accessible, stable, and powerful mobile hosts to relay invalidation information for other cache nodes.

The protocol periodically calculates relay-peer selection criteria using three parameters:

- coefficient of peer access rate,
- coefficient of the stability of a node, and
- coefficient of the energy level of a node.

As Figure 3 shows, a cache node becomes a relay-peer candidate if its coefficients satisfy a criterion.[10] A candidate in turn becomes a relay peer if it can listen to the invalidation messages sent from the source node of its cached data item—the candidate is within a distance less than a certain number of hops from the source node—and successfully obtains the approval message from the source node.

The source node and relay peers form a relatively stable overlay in which the source node pushes data and invalidation information to its relay peers periodically while communication between cache nodes and relay peers is pull-based to deal with unreliable network connections. Thus, if a cache node has been disconnected from the network, upon reconnection it need only poll the nearest relay peer instead of polling the source node to check the status of its cached data items.

Simultaneous and asynchronous push and pull operations reduce query latency and communication overhead. Moreover, the use of relay peers makes it possible to adaptively handle query requests with different consistency requirements. Also, as a hybrid of SF and SL approaches, RPCC need only store relay peers' information on the source node, which saves considerable memory and simplifies database management.

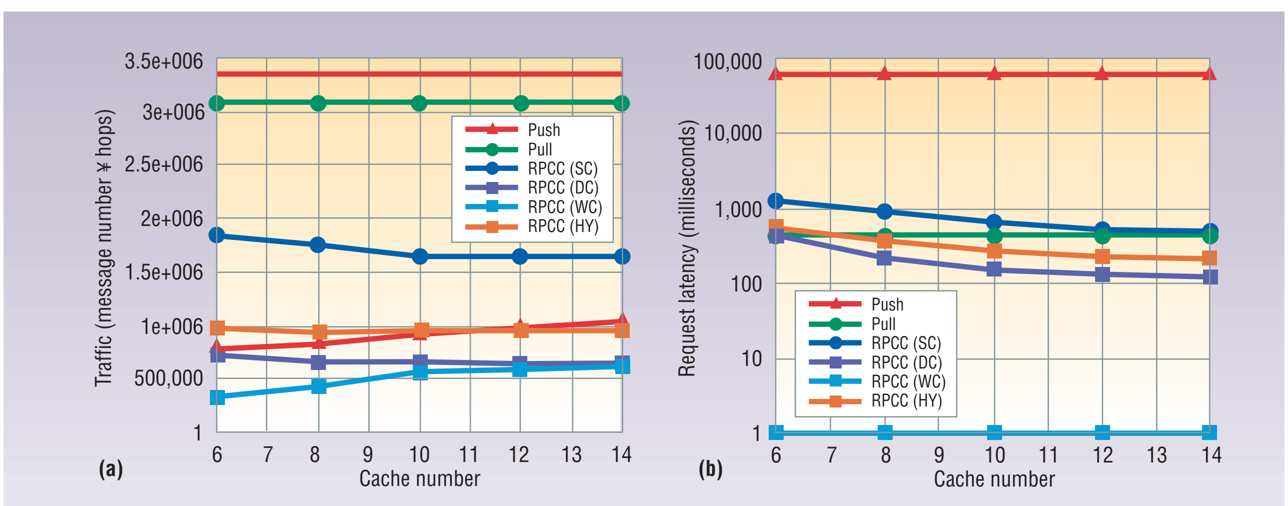

Figure 4. Performance comparison. RPCC leads to (a) less network traffic than a push-based approach and (b) lower query latency than a pull-based approach.

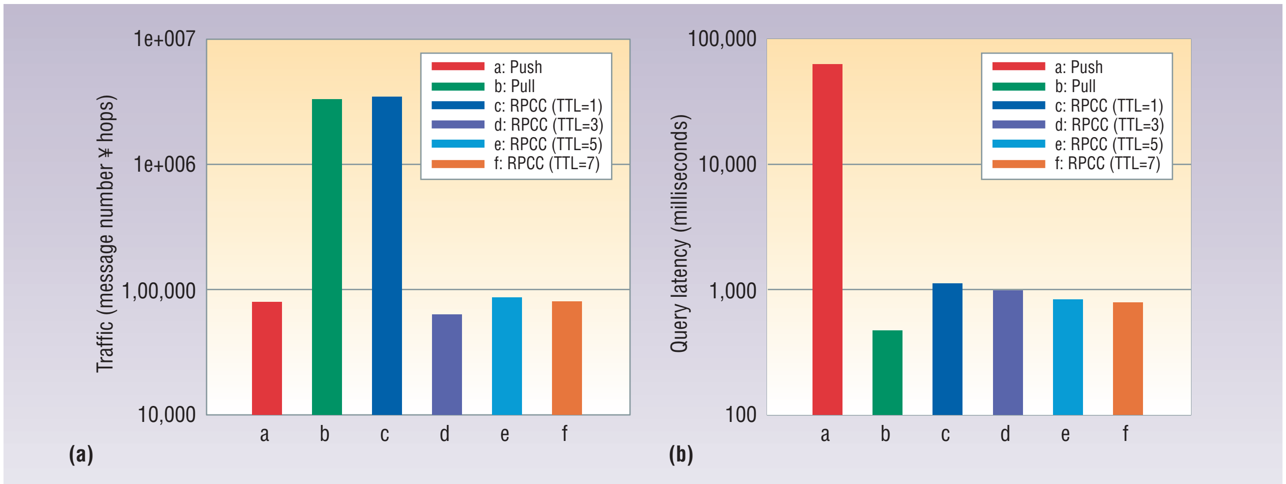

Figure 5. TTL value. Impact on (a) network traffic and (b) query latency.

To assess RPCC's effectiveness versus push- and pull-based schemes, we used the Global Mobile Information Systems Library (http://pcl.cs.ucla.edu/projects/glomosim) to simulate a mobile ad hoc network with 50 cache peers randomly distributed over a 1,500 × 1,500 meter area. Each mobile host generated an independent stream of updates to its source data and query requests at exponentially distributed intervals. Table 1 lists the other simulation parameters.

We evaluated six cache consistency maintenance strategies on the network: push, pull, and RPCC with strong, delta, weak, and hybrid consistency, in which requests with three different consistency requirements had equal probability. Figure 4 compares their performance with various cache numbers and clearly indicates that RPCC leads to less network traffic than a push-based approach and generates lower query latency than a pull-based approach.

RPCC is a generic and flexible approach that makes it possible to meet the specific requirements of different applications by setting the time-to-live value of invalidation messages from the source node. As Figure 5 shows, if the TTL value is small so that too few relay peers are available, the system is more likely pull-based with higher network traffic and lower communication latency; if the TTL value is large enough that most of the cache nodes can be relay peers, the system will more likely be push-based.

Our 3D framework facilitates the design, classification, and evaluation of current cache invalidation strategies in both single-hop and multihop mobile networks, while our proposed cache consistency strategy effectively separates the cache invalidation procedure into two subprocedures that can be carried out asynchronously and simultaneously. Our future work will examine the formation of relay-peer overlays and address the design of adaptive cache invalidation strategies, moving beyond a static combination of the three design options and giving application designers the dynamic capabilities to customize strategies at runtime.

Acknowledgments

This work was partially supported by the University Grant Council of Hong Kong under CERG grant B-Q937 (PolyU 5105/05E), the Hong Kong Polytechnic University under ICRG grant G-YE57, and the 973 Program of China under grant 2002CB312002.

References

1. T. Imielinski and B.R. Badrinath, "Mobile Wireless Computing: Challenges in Data Management," *Comm. ACM,* vol. 37, no. 10, 1994, pp. 18-28.
2. D. Barbará, "Mobile Computing and Databases: A Survey," *IEEE Trans. Knowledge and Data Eng.,* vol. 11, no. 1, 1999, pp. 108-117.
3. G. Cao, L. Yin, and C.R. Das, "Cooperative Cache-Based Data Access in Ad Hoc Networks," *Computer,* Feb. 2004, pp. 32-39.
4. L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," *Proc. IEEE Infocom 2004,* IEEE Press, 2004; http://mcn.cse.psu.edu/paper/yin/infocom04.pdf.
5. H. Shen et al., "Energy-Efficient Caching and Prefetching with Data Consistency in Mobile Distributed Systems," *Proc. 18th Int'l Parallel and Distributed Processing Symp.,* IEEE CS Press, 2004, pp. 67-76.
6. J.H. Howard et al., "Scale and Performance in a Distributed File System," *ACM Trans. Computer Systems,* vol. 6, no. 1, 1988, pp. 51-81.
7. J. Gwertzman and M. Seltzer, "World-Wide Web Cache Consistency," *Proc. Ann. Usenix Tech. Conf.,* Usenix Assoc., 1996, pp. 141-151.

8. M.N. Nelson, B.B. Welch, and J.K. Ousterhout, "Caching in the Sprite Network File System," *ACM Trans. Computer Systems,* vol. 6, no. 1, 1988, pp. 134-154.
9. C. Gray and D. Cheriton, "Leases: An Efficient Fault-Tolerant Mechanism for Distributed File Cache Consistency," *Proc. 12th ACM Symp. Operating Systems Principles,* ACM Press, 1989, pp. 202-210.
10. J. Cao et al., "Consistency of Cooperative Caching in Mobile Peer-to-Peer Systems over MANET," *Proc. 3rd Int'l Workshop Mobile Distributed Computing,* IEEE CS Press, 2005, pp. 573-579.

***Jiannong Cao** is a professor in the Department of Computing at Hong Kong Polytechnic University, where he also directs the Internet and Mobile Computing Lab. His research interests include parallel and distributed computing, networking, mobile computing, fault tolerance, and distributed software architecture and programming. Cao received a PhD in computer science from Washington State University. He is a senior member of the IEEE and the ACM. Contact him at csjcao@comp.polyu.edu.hk.*

***Yang Zhang** is a PhD student in the Department of Computer Science and Engineering at the Pennsylvania State University. His research interests include mobile computing and networking. Zhang received an ME in computer science from Nanjing University, Nanjing, China. He is a student member of the IEEE. Contact him at zhang@psu.edu.*

***Guohong Cao** is an associate professor in the Department of Computer Science and Engineering at the Pennsylvania State University, where he also directs the Mobile Computing and Networking Laboratory. His research focuses on wireless networks and mobile computing. Cao received a PhD in computer science from the Ohio State University. He is a member of the IEEE. Contact him at gcao@cse.psu.edu.*

***Li Xie** is a professor of computer science at Nanjing University, and formerly served as both vice president of the university and dean of the Department of Computer Science and Technology. His research interests include parallel and distributed systems, secure operating systems, and software engineering. Zhang received a BSc in computer science from Nanjing University. Contact him at xieli@nju.edu.cn.*