Volume 6, Issue 5, May 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Cache Invalidation Method Scheme for Mobile Ad-hoc Networks

Aparajita Nailwal, Mukul Varshney, Shivani Garg
Department of CSE, Sharda University, Greater Noida,
Uttar Pradesh, India

Abstract—Wireless bandwidth and battery power can be conserved by caching frequently accessed data by mobile clients , at the expense of some system resources to maintain cache consistency. In literature, several mechanisms have been proposed to address the problem of cache consistency in cellular wireless networks. In this paper special interest is with Database applications in wireless devices .Due to low bandwidth in the wireless environment and frequent disconnects from the server they are different from traditional client server database applications . Hence, efficiently caching some of the elements that are frequently required by the mobile device can save the scarce resources and give better response time for the client application .This paper also discuss about some of the mechanisms that have been proposed by researchers in the direction of data consistency maintenance between the clients and the data servers.

Keywords— cache consistency, data consistency, scarce resources, database applications, data servers.

I. INTRODUCTION

Homes, Telecommunication networks and Enterprise installations avoid the costly process of introducing cables for building a connection between various equipment locations using wireless networking [1]. The advances in hardware and wireless technologies such as Wi-Fi and the proliferation in the usage of mobile devices have made wireless networks ubiquitous. The falling cost of both communication and mobile has made mobile computing commercially affordable to both business users and private consumers [1]. In the near future, people with battery powered mobile terminals (MTs) can access various kinds of services over wireless networks at any time or any place. Transmission over wireless channel is expensive due to limited bandwidth, transmission latency and large energy consumption by mobile clients.

This paper focuses on the database applications running on mobile devices. The client-server paradigm in the wireless environment differs significantly from the traditional approach because of two main reasons: frequent disconnections and low bandwidth. Also, in a mobile environment, upstream queries (i.e., from client to server) are more resource-consuming than the downstream queries (i.e., from server to client), so there is a need to reduce the number of trips made to the server. Caching data items at the client side is a solution. However, the caching techniques used for traditional database models cannot be applied in this arena as it is. Additionally, mobile clients also have limited resources like power, so any of the caching schemes have to be energy efficient and support long and frequent disconnections. In the following paragraph, we look at the criteria that must be considered for designing a cache management mechanism for mobile environment.

Caching frequently accessed data, is an important technique commonly used to reduce these costs, Cache invalidation strategy is used to ensure that the data items cached by a mobile client are consistent with those stored on the server [9]. In a Wireless network, data caching is essential as it reduces contention in the network, increases the probability of nodes getting desired data, and improves system performance. The major issue that faces cache management is the maintenance of data consistency between the cache client and the data source [11]. All messages sent between the server and the cache are subject to delays, thus, impeding consistency by download delays, that are considerably noticeable and more severe in wireless mobile devices. Client caching means that each client caches some of its received data items in the local cache [12]. If a new query arrives at a later time, requesting for the same data, the cached copy can be used without going through the server again. This reduces the amount of uplink traffic, possibly leading to reduce bandwidth requirement. In practice, data items are updated from time to time in an asynchronous manner [9]. A client cannot rely on its cached copies to answer queries forever. Instead, the validity of the caches should be first ensured. All cache consistency algorithms are developed with the same goal in mind to increase the probality of serving data items from the cache that is identical to those on server [11]. However, achieving strong consistency, where cached items are identical to those on the server, requires costly communications with the server to validate (renew) cached items, considering the resource limited mobile devices and the wireless environments they operate in. Wireless networks are classified as Infrastructure based and ad hoc based[1] .

II. CACHE INVALIDATION METHODS FOR AD-HOC NETWORKS

All Mobile nodes in infrastructure based networks directly contact with the Access points. So, the cache consistency is easier in Infrastructure based networks. But in Ad-hoc networks some mobile nodes cannot directly contact with the Access points. The Cache consistency is difficult in Ad-hoc networks.

A. COACS (A Cooperative and Adaptive caching system)

COACS is a distributed caching scheme. Cache consistency is difficult in Ad-hoc networks compared to the Infrastructure based networks [12]. COACS introduces new architecture for cache consistency. COACS consists three types of nodes: Requesting node (RN), Query directory (QD), Caching node (CN). A QD's task is to cache queries submitted by the requesting mobile nodes, while the CN's task is to cache data items (responses to queries). As shown in Fig. 1, RN sends request, QD forwards that request to the CN. CN checks whether the requested data available or not. If the requested data is available then it forwards reply to the RN. If the CN does not have the requested data then request is forwarded to the Server.

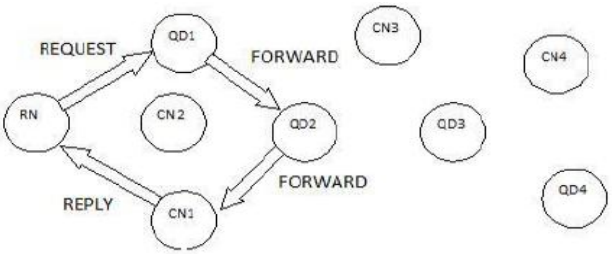

Fig 1: A Cooperative and Adaptive caching system

B. DCIM (Distributed cache invalidation mechanism)

DCIM is a client based cache consistency mechanism that implements adaptive time to live (TTL), piggybacking, and prefetching, and provides near strong consistency capabilities [11]. DCIM follows the COACS Architecture. As shown in Fig. 4, Requesting node (RN) sends request using Data request packet (DRP) packet. Query directory (QD) checks whether the requested data is available in caching node (CN). Then, CN returns reply using Cache update request (CURP) **[11]**. Data items are assigned adaptive TTL values that correspond to their update rates at the data source, where items with expired TTL values are grouped in validation requests to the data source to refresh them. This validation requests send using CURP.

Server gives validation reply using Server update data (SUDP) and Server validation reply (SVRP) [11].whereas un expired ones but with high request rates are prefetched from the server. Since COACS did not implement a consistency strategy, the system DCIM (Distributed Cache Invalidation Method) provides the several improvements: Enabling the server to be aware of the cache distribution in the MANET, Making the cached data items consistent with their version at the server, Adapting the cache update process to the data update rate at the sever relative to the request rate by the clients, With these changes, the overall design provides a complete caching system in which the server sends to the clients selective updates that adapt to their needs and reduces the average query response time.

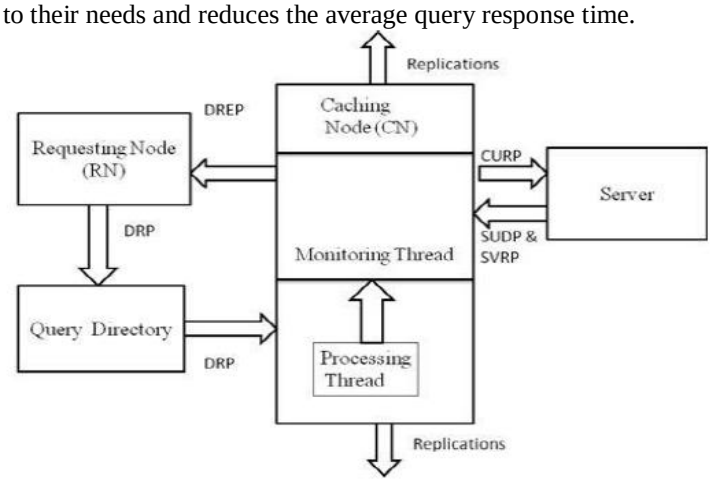

Fig 2: Distributed cache invalidation mechanism

C. SSUM (Smart server update mechanism)

SSUM is a server-based approach that avoids many issues associated with push-based cache consistency approaches. In SSUM, the server autonomously sends data updates to the CNs, meaning that it has to keep track of which CNs cache which data items **[**14]. This can be done using a simple table in which an entry consists of the id of a data item (or query) and the address of the CN that caches the data.

A node that desires a data item sends its request to its nearest QD. If this QD finds the query in its cache, it forwards the request to the CN caching the Item, which, in turn, sends the item to the requesting node (RN) [14]. A Server sends update reports frequently to the CN's for indicating updation in the cached data items [14].

Table 1: Comparison Chart of Catch Invalidation Scheme

	Latency	**Network utilization**	**Good performance during Disconnections**	**Message Traffic**
COACS	No	Yes	No	Yes

DCIM	No	Yes	No	Yes
SSUM	No	No	No	No

III. CACHING OVERVIEW

A caching mechanism is characterized by its caching granularity, cache coherence strategy and cache replacement policy [6]. Caching granularity answers the basic question of what to cache; this includes the decisions made by the application to cache right kind of information so that it can optimize the cache storage. Had the storage capacity of a mobile device been unlimited, the application could have cached all the data items that it requested from the server. Since, this is not the case, intelligent decisions made by the application about what to cache, in the very beginning, affects the efficiency of caching system markedly.

After the decision has been made as to what to cache, the next step is to define a cache coherence strategy whose function is to keep the cache consistent. A number of different schemes have been proposed in this area. Majority of them are cache invalidation schemes which are based on push-based mechanism, in which a server repeatedly broadcasts/multicasts data reports whenever a specific item changes in the database. Listening to such reports, a client can decide if the cache it is keeping, is valid or not and can mark the specific items to be invalid.

IV. CACHING GRANULARITY

Caching mechanisms in conventional client-server environment are usually page-based, primarily because the server's storage is page-based and transmission bandwidth is comparable to the disk bandwidth. The overhead of transmitting one item or a page is same. Furthermore, the overhead will be paid off if any item within the transmitted page will be accessed by a client in the near future. In general, page-based caching mechanism requires a high degree of locality among the items within the page to be effective. In contrast, mobile clients are powered by short-life batteries. Caching a page will result in wasting of energy when the degree of locality is low. Downloading a whole page instead of just the requested item will result in long query delay, so page based caching is not suitable for mobile environment. Other techniques like attribute caching etc., lowers the degree of granularity and hence are more efficient In the following sub-sections we explain these techniques and compare them.

A. Object Caching

The granularity level in this scheme is individual objects. Each mobile client tends to have its own set of hot objects that it accesses most frequently. Furthermore, a client might access different attributes of an object in different queries it initiates [6]. Using this scheme, whenever the server receives a request, it sends all attributes of a qualified object, x. The idea here is to pre fetch all attributes of a requested object to the client that will cache the returned attributes [11].

B. Semantic Caching

The idea of Semantic Caching (SC) in mobile environment stems from the fact that semantics of cache content can be exploited to make better decisions about what to cache [4]. The mobile client maintains the semantic descriptions and results of previous queries in its cache. If a new query is totally answerable from the cache, then no communication with the server is necessary [8]. If it can be partially answered then the original query is trimmed and the trimmed part is sent to the server and processed there. By processing queries this way the amount of data transferred over the wireless link can be substantially reduced. Table 2 below, compares advantages of semantic caching to traditional Page Caching (PC). In semantic caching, cache is divided into fragments and sub-fragments. After a query is evaluated by running it on the first fragment, the remainder query can then be checked against the remaining cache and trimmed by next candidate fragment or sub-fragment. This process continues until there is no fragment that could contribute to the query result. The final remainder query is sent to the server for processing.

Table 2: Semantic Caching VS Page Caching

Item	PC	SC
Cache space overhead	High	Low
Coherency Control	Ineffective	effective
Communication Cost	Expensive	Cheap
Disconnection Handling	Inefficient	effective

V. COMPARISON

Firstly, both object and attribute caching are defined for object-oriented databases [6]. Given an object-oriented database, although the decision to choose between object-based or attribute-based caching mechanism is more application specific, it is noteworthy that page-based caching granularity does not give good results in a mobile environment. To reap the benefits of both objects and attribute caching, a hybrid model can be employed. Applications like traffic information systems should employ a hybrid model as based on the location, query results can vary significantly. In addition the hybrid model can be combined with semantic caching to gain performance benefit. Semantic caching mechanisms can also benefit applications that involve answering continuous queries [9] like finding the nearest restaurant in the vicinity of a mobile car. However, when the query content is huge, semantic caching may not give the anticipated benefit.

VI. CONCLUSION

The inherent limitations of mobile computing systems present a challenge to the fixed problems like consistency, concurrency and currency of data to client systems. The amount of research in this area in the last few years has been staggering. However, some problems remain open for research. There is a need for better protocols in the area of data sharing, message delays, duplication of data at client caches and transaction management, Better interfaces, clever algorithms that exploit locality to shape the answers to queries.

A lot of mechanisms have been proposed which depend upon invalidating the data based on timestamps, lately there has been emphasis on designing cache coherence strategies depending upon needs of set of applications like applications that depend upon location dependent data. Cache replacement policies devised for mobile databases take into consideration the access patterns to conserve the storage space which is a scarce commodity in a mobile device.

We believe that the area of cache replacement has not received much attention. Lot of\ work needs need to be done in this area, especially in case of location dependent data, to find better replacement policies. Work is also needed in area of cache granularity to better predict which attributes to cache and also to move beyond object-oriented databases. In the future work, we should be able to answer which combination of techniques can co-exist to meet the needs of particular set of applications and if any of these algorithms are implement in commercial mobile applications.

REFERENCES

[1] Jochenschiller, ―Wireless communication‖, Addison-Wesly, 2003.
[2] Daniel Barbara, Imielifiski, "Sleepers and Workaholics: Caching Strategies in Mobile Environments" , Proc. OfACM SIGMOD, May 1995
[3] Gupta S.K.S, San Jose, Srimani P.K, "A strategy to manage cache consistency in a disconnected distributed environment", Parallel and Distributed Systems, IEEE Transactions Vol 12, 2001."
[4] Wang Z, Arlington TX, Das S.K, HaoChe, Kumar M, "A Scalable Asynchronous Cache Consistency Scheme (SACCS) for mobile environments", Parallel and Distributed Systems, IEEE Transactions, 2004.
[5] Jin Jing, Ahmed Elmagarmid, Abdelsalam (Sumi) Helal, Rafael Alonso, "Bit-Sequences: An adaptive cache invalidation method in mobile client / server environments", Springer, Mobile Networks and Applications, Vol 2, 2006.
[6] Yu Du and S.Gupta, ―COOP-A Cooperative Caching service in MANETs‖, Proceedings of the IEEE ICAS/ICNS (2005), pp.58-63, 2006.
[7] Hong V. Leong, Antonio Si "On Adaptive Caching in Mobile Databases" April 1997, Proceedings of the 1997 ACM symposium on Applied computing
[8] Alok Madhukar, Reda Alhajj "An Adaptive Energy Efficient Cache Invalidation Scheme for Mobile Databases" April 2006, Proceedings of the 2006 ACM symposium on Applied computing SAC '06
[9] Ken. C.K. Lee, H.V. Leong, Antonio Si "Semantic Query Caching in Mobile Environments", June 2005, Proceedings of the 4th ACM international workshop on Data engineering for wireless and mobile access
[10] Chi-Yin Chow, Hong Va Leong, Alvin T. S. Chan "Distributed Group-based cooperative Caching in a Mobile Broadcast Environment" May 2005, Proceedings of the 6th international conference on Mobile data management MDM
[11] Boris Y. L. Chan, Antonio Si, Hong Va Leong . "Cache Management for Mobile Databases: Design and Evaluation." 1998, Proceedings of the Fourteenth International Conference on Data Engineering
[12] Jianliang Xu; Qinglong Hu; Wang-Chien Lee; Dik Lun Lee; "Performance evaluation of an optimal cache replacement policy for wireless data dissemination" 2004, IEEE Transactions on Knowledge and Data Engineering
[13] Quen Ren, Margret Dunham, "Semantic Caching in Mobile computing" (2001)
[14] Baihua Zheng, Dik Lun Lee, "Semantic Caching in Location-Dependent Query Processing", Lecture Notes in Computer Science (2001)

