Classification as per Dewey System: 600 Technology, Applied Science.

Asynchronous Distributed-Based Cache (ADBC) to Achieve Consistency in Server/Client Mobile Networks

Hatem Hamad, Associate Professor, IUG, hhamad@iugaza.edu.ps,
Hanan A. Thuraya, Computer Department, IUG, habuthuraya@ gmail.com,
and Huda Hubboub, Computer Department, IUG, hhubboub@ gmail.com.

Abstract - A practical approach for enhancing performance in a mobile environment is caching frequently accessed data items on the client side. In a wireless system it may be intricate to manage the data at cache while sustaining consistency due to various limitations. One attractive consistency scheme to overcome such limitations is invalidation reports (IRs), even though IRs suffers from the query delay since the user has to verify the data first and if it is invalid he has to obtain it from the server. In this paper we propose an IR-based invalidation algorithm which can significantly reduce query delay and the uplink per query. Our model efficiently utilizes the property of sharing data between multiple users and employs a distributed cache scheme to alleviate some load from base station (BS) related to processing and maintenance and to fairly distribute it among interested mobile users (MUs). This will enable us to capture the effects of reducing query delay while maintaining the previously achieved requirement of bandwidth utilization and energy conservation. Detailed analytical analysis and simulation experiments are carried out to evaluate the proposed methodology and to demonstrate that our goal has been achieved by the observed significant drop in the delay.

1. INTRODUCTION

The mounting demand for wireless technology and related applications has prompted technology companies to invest heavily to introduce a wide range of wireless products such as laptops, cellular phone, etc., to meet customers' demand while preserving high efficiency and data integrity that suit the needs of a broad range of MUs. Ideally, a MU should be able to have access to desired information such as news, financial information, stock prices, etc. whenever and wherever he desires. However, the mobile environment is different and faces two major limitations, namely the system has limited bandwidth, and the mobile user is restricted by limited power device. Hence, caching can become a viable technique to support effective service by having a local copy of the data at the user's terminal. However, if the has been changes with the original copy of the data, the local copy at MU will not be valid any more requiring the MU to verify the validity of the data to achieve consistency before responding to a query. Other features of the mobile environment that could also impede achieving high consistency is the frequent disconnections of MU which

can either be voluntarily switching off to conserve battery or involuntarily due to failure or roaming. Here, caching is also viewed as a good technique that would improve the system performance by reducing the query delay.

Whenever caching is used, there must be an arrangement to deal with the issue of frequent modification of the data at the source, which is called cache consistency. This consistency can be achieved using three basic strategies. The first one is time-to-live (TTL) based used in many Web caches where a TTL bit is used to assign the validity of data to a specific time, the MU initiates to get the fresh copy of the data when TTL reaches zero. In the second approach the MU has to poll every time-like TTL=0-, however this insures the validity of data albeit at the expense of high overhead compared with the first approach where some degree of data inconsistency is tolerated. The invalidation report based IR is the third strategy where the users are informed about the invalidity of the data so they can't use their copy.

In this paper, we will address the problems associated with the IR based cache strategies by presenting a caching scheme for mobile/client wireless network. The proposed approach employs asynchronous distributed based-cache using invalidation report to sustain cache consistency. Our scheme efficiently utilizes the property of sharing data, i.e. there must be a reasonable number of MUs who have common interests, consequently requesting the same data. Each MU has his own local cache, and once he requests a particular data, the BS delegates the tasks of this data maintenance to that MU and assigns him to be user in charge (UIC). Our interest is to attain consistency of data at UIC, because if any other MU asks for that data, the BS will redirect him to the UIC, where he can obtain it through a separate channel. The BS has to play as a mediator between users sharing the same data. The heavy burden of handling all the data (processing-maintenance) is shifted from the BS and distributed in a fair manner between MUs. The BS has an entry for each data item cached by its users. Here, we add other bits for the purpose of specifying the validity of data, such as the ID of UIC, ID of successor, the time of the last event relevant to data; flag bit is set to 1 when the BS is sure that UIC has a fresh copy of the data, and the common TTL bit. Our model primarily aims to improve the system performance by increasing the through put while reducing the query delay, number of uplink requests, and performing good utilization of limited bandwidth. The rest of paper is structured as follow: section 2 contains brief description of the related work, while the system model and assumptions are introduced in section 3. Both formal and brief description of our caching scheme is presented in sections 4 and 5 respectively. Section 6 gives the performance analysis of the proposed scheme in terms of hit ratio and

query delay, in addition it reports on comparative simulation results of ours and existing techniques and finally the conclusions of our paper are drawn in section 7.

2. RELATED WORK

Existing IR schemes can be classified into stateless and statefull. The basic ***stateless*** cache invalidation scheme was first proposed by Barbara and Imie linski, where an IR is broadcasted by the server to point out data items that are recently updated by the source. The MU has to continue listening to wireless channel until receiving the IR in order to invalidate the corresponding data. Based on IR, different approaches have been proposed, a bit sequence algorithm is employed by maintaining information about half of data items recently updated, and then MU with long disconnection period can invalidate all his cache. However longer IR messages have to be broadcasted. All based-IR techniques suffer from long delay query since the MU must wait to the next IR to answer the query and ensure cache consistency. To address the problem, many solutions have been suggested in order to reduce the waited time spent by MU, Hu and Lee choose to broadcast reports based on the update frequency of the data and query rate / disconnection period of MU. While Cao deals with the problem by inserting updated invalidation report (UIR) into each IR interval. UIR only contains the data items that have been updated after the last IR has been broadcasted. Yuen assigns a validity interval (AVI) to the data, and then the MU can verify its data by comparing the last update time plus its AVI. IR is broadcasted on AVI changing to allow MU to invalidate the related data. The attractiveness of aforementioned stateless approaches can be attributed to three reasons. First, their scalability regardless of the number of MUs listening to IR. Second, the IR has a fixed size independent of the clients' number. The third reason is that they can be power-efficient models since MUs have the opportunity to switch to sleep between two IRs. On the other hand, the main drawback is the fact that although all modifications added in the previous algorithms to IR, there is still a time wasted by MU while waiting for IR besides the time spent after requesting the updated version from the server before receiving it.

The statefull scheme surmounts the drawbacks associated with the stateless one, however only few literatures tackled this scheme. Kahol proposed a model where the mobile switching station (MSS) records all data stored in its MU caches and whenever MSS receives IR about specific data item, it immediately broadcasts the message; hence the concerned MUs can follow up the validity of their caches. This model also deals with the disconnection problem, since MSS has per user home location cache (HLC), and when awake the MU can explain about the state of its data from its own HLC. The positive aspect of this scheme

is that only interested MUs capture the message, thus non-interested users does not waste power receiving the message. The negative aspects can be summarized in high uplink signaling arising from ACK to confirm that the user gets the IR and the storage capacity of per user HLC, although there is shared data between users. Therefore, the efficiency of this model heavily depends on the number of user connected to each MSS. To address this problem, a novel cache consistency maintenance algorithm Scalable Asynchronous Cache Consistency Scheme (SACCS) was proposed, where a cache is located at MSS, and in order to perform consistency only a flag bit is added to the cache of both MU and MSS. The positive aspect of SACCS is its reduced query delay compared with other previous algorithms. While its aspects can be revealed in two points. First, intense burden will be on MSS either in processing and maintaining data. Second, its handling with roaming since MU has to invalidate its cache when traveling to a new cell regardless of his state. If the MU was awake, consequently he is sure about the validity of the data. A development for SACCS has been made to Dynamic SACCS by extending the research to multi-cell system. The idea behind this scheme is to attain mobile data objects globally or locally in order to achieve minimum consistency cost. The cost is considered with relation to the update frequency of each data objects, MUs' access pattern, roaming frequency, and other criteria. Many other literatures address this problem with different issues; the idea of strong and weak data consistency is applied by Vora and Tari, where consistency is maintained by using both strict and tolerant read/write time locks that enable data sharing for a fixed time period. Unlike others, Lai proves the validity of IR in cache consistency analytically and a distinction between the existing algorithms is explained in details showing the weakness and strengthens in each scheme.

3. SYSTEM MODEL

Prior to describing our ADBC model, a brief description of the wireless mobile architecture considered in this paper will be presented. Fig.1 provides illustrates the proposed system which consists of multiple servers, BSs and MUs where the MU can access the data at the server via the BS. Each BS is connected to the servers through wired links while serving its MU via wireless network. The MU has its own local cache, where he stores his relevant data while BS maintains the cache contents of its MUs using its cache, this cache has only identifiers of the data which the MU having the recent copy of that data. The system is read-only meaning that all updates are performed at the server and MU does not perform any updating process.

4. A BRIEF DESCRIPTION OF THE PROPOSED SCHEME.

4.1 Invalidation Report:

IR is a strong technique to achieve consistency, where the MU is informed about the invalidity of the data when it has been updated using different schemes. Existing IR schemes can be classified into stateless and statefull. In a stateless server approach, the server is not aware about the contents of its mobile users, and the MU has to verify the validity of the data before answering the query or using other strategies. In a statefull server approach, the server is aware about the contents of its mobile user and would inform them about any modification. Each approach has its advantages and disadvantages. The stateless approach has simple data base management. However, it results in a heavy traffic due to connection between the mobile and server whenever a query occurs, it is not scaled well, it has a poor support to disconnections and mobility and the MU has to keep listening to the channel to determine the state of its cache resulting in a waste of power. The statefull approach is scalable, but it provides an overhead due to server management and implies a restriction on the movement of MU, since it has to notify the server whenever he moves to another cell.

in our model, if the data at the server has been modified, the server informs the BS via wired links that the corresponding data is no longer valid and accordingly the BS has to identify the invalidity of data by clearing the flag bit at UIC of the data to zero, then it broadcasts the IR, where all the users receiving the IR have to invalidate their corresponding data. Our main focus is to sustain the consistency of data at the UIC since he is the reference for that data. The UIC has to ask for the new version of the data from BS, which brings it from server and broadcasts it. If the UIC receives it correctly, he will send ACK, and immediately the BS set its flag to 1.

4.2 A query Request

Whenever a query occurs, the MU explores his cache searching for the required data, if a hit occurs, he answers the query, otherwise he sends a message to the BS requesting that data. Upon receiving the request, the BS looks for the requested data, and if none of its users has a copy of it, the data will not be readily available in its cache and it must ask the server to send that data through wired network. Upon receiving the data, the BS resends it to the interested user and records its item and the id of the MU as if the BS delegates the control of this data to the MU naming him as UIC. Thus, the BS will redirect any user who asks for this data to the corresponding UIC. In addition, the BS will replace the current UIC by that user, and the old UIC will be the successor, a flag bit is used to assign the state of the data at UIC. The data path from UIC to the requester yields a smaller delay which is our concern in our scheme. This differs from other previous IR-based algorithms which concentrate on solving the problem associated with long disconnections.

4.3 Roaming

Before moving to a new cell, the MU has to notify the BS about his intention of moving and his new location. Then, the BS can either shift the data under the MU's charge to the corresponding successors, or alternatively sends the requests to him at his new location. While the MU is traveling to a new cell, he may be either asleep or awake. In the sleep state the MU has to invalidate his cache upon waking up, however, if he was awake during roaming, then he is certain about the validity of his data and can keep it. Also he will be able to provide the new BS with his data and makes it accessible whenever needed.

5. A FORMAL DESCRIPTION OF THE PROPOSED SCHEME

5.1 Data Structure

As illustrated in Fig.2, each data has a unique identifier i, where the notation di will be used to denote the data object for identifier i. Data structure maintained for every entry in base station cache **(BSC)** contains ***(i ,ti ,ICi, Si , Flag, TTL,*** **and** ***NBS*****)**, As for the MU, the data for each entry in the mobile user cache **(MUC)** contains **(*i, di, ti, V-Flag, IC-flag* and *TTL*)** , these structures are organized as shown in Table 1.

5.2 System Messages.

The communication messages of our scheme can be defined as follow:

INVALIDATION REPORT( ): When the data is updated at the source, the server sends IR to all BSs, which in turns broadcasts the IR to inform MUs to invalidate their corresponding data, if the MU is UIC, he has to Acknowledge the BS and request the modified data.

DATA REQUEST( ): This message is sent by the MU to his BS demanding the data item i either if di is not found in the local cache, or if the existing copy is invalid.

VERIFY DATA ( ): Sent by MU upon waking up to verify the state of its cache, since there is a probability that he missed IRs during his sleep.

DATA ( ): Broadcast by BS to all MUs to update their data, when any request for the data is received by BS from any MUs.

CONTROL MESSAGES ( ): Represented by ACK sent from MU to the BS to confirm the delivery of data, and allowing it setting its flag to the appropriate values.

5.3 System Algorithm:

Two algorithms are presented for ADBC namely **BS-main( )**, and **MU-main( )**, the first one is to handle the MU query request, while the other is to demonstrate the way of transferring data and maintaining

consistency between MU and BS. In **BS-main ( )**, as BS receives *invalidation-report ( )*, it broadcasts IR ( ) to all MUs and reset flag bit while waiting for ACK from UIC in order to set the flag bit. Upon receiving data-*request ( )* message from the MU, it checks local cache, when hit, the BS fetches for UIC, and inform MU about UIC, otherwise (when miss), it broadcasts the queried data from the server to all MU's, when receiving *verify-data ( )* message, it decides the validity of data by comparing the time stamp of that item in his cache with the time stamp sent by MU's, then a confirmation message is broadcasted. In **MU-main ( )**, whenever query is received, and MU does not find the data in his local cache, he sends *request-data ( )* message to the BS. When receiving *IR ( )*, set V-flag and update all entries. When *receiving data ( )*, set V-flag and IC flag. When waking up, MU sends *Verify-data ( )* to guarantee the validity of its data, in case that he miss IR during his sleep.

6- PERFORMANCE ANALYSIS:

In this section, we will investigate the performance of our proposed scheme, we derive the analytical model, then the simulation is performed using **ns-2** for a system model of a server, one base station and varying number of MUs. The purpose of this evaluation is to compare our scheme with AS and SACCS schemes previously proposed, based on the mean query delay (D_{avg}) and uplink request per query U_{req}, the following assumption are made in our model:

- The data base is of size M, and can be updated only by the server, where mobile hosts perform only read query without any modification to that data.
- Each mobile user generates a read only query according to Poisson distribution with mean rate λ_g .
- Each data item's update process follows an exponential distribution with mean $1/\mu_D$.
- The cache replacement scheme used by mobile user is least recently used (LRU) in the case of no invalid data in his cache.
- MU alternates between sleep and awake states; a factor s represents the time spent in sleep mode.

First, we estimate the average hit ratio at both the local cache of a MU and the BS, while the average delay between MUs query uplink is obtained by simulating the proposed model analytically using ns-2 for a varying number of mobile users.

6.1 Estimation of Miss Ratio at Local Cache:

Due the fact that all queries interrupt the MU during his disconnection are eliminated, then the effective query rate can be expressed as $\lambda_e = (1-s)\lambda_g$,While the rate of generation a query for a specific data item i is given by the relation:

$$\lambda_i = \frac{\lambda_e}{M} = \frac{(1-s)\lambda_g}{M}$$

A query for a data i would be a miss in the local cache of the mobile node -and accordingly, up link request

$$P_{local-miss} = \int_0^\infty \lambda_i e^{-\lambda_i t}.(1-e^{-\mu_d t})dt \quad = \frac{\mu_d}{\mu_d + \lambda_i} = \frac{M\mu_d}{M\mu_d + (1-s)\lambda_g}$$

is necessitated – in the case of receiving IR, while the new valid copy of this data has not been obtained yet, then the probability of miss due to this event is given by:

6.2 Estimation of Miss Ratio at Base Station:

For simplicity, we suppose BS-miss to denote the case that specified data is not available at any MU cache, since the BS can indicate whether this data exists in another user's cache. BS miss occurs if no user queries the data item i during T interval, which in fact is deeply related to the ratio of shared data ***d.***

$$P_{BS-miss} = e^{-\lambda_g * d * (n-1)T}$$

That is the probability of missing the data is one if no shared data is found at the database, or if only a unique user is served by the network.

6.3 Valuation of Query Delay:

NS-2 simulator is used for estimation of the query delay considering the effect of changing the number of MUs. To achieve fairness in the assessment of the average query delay, the system is modeled in two scenarios as follows: 1) only unique user requests the data from the base station/server and 2) all users ask for the data. The obtained results are summarized in table 2.

Based on the previous estimations of $P_{local-miss}$, $P_{BS-miss}$, t he average delay experienced by any query in the system is given by the equation:

$$Td = P_{local-miss}(D_{MU-BS} + P_{BS-miss}D_{DB-BS}) + P_{local-miss} * D_{MU-UIC} * (1 - P_{BS-miss})$$

Where D_{MU-BS} , D_{MU-DB} and D_{MU-UIC} are the delay required by the user to bring the date from base-station, server, and user-in-charge respectively

7. PERFORMANCE COMPARISON:

7.1 Experimental Setup:

We simulated our ADBC proposed invalidation scheme using the parameters depicted in Table 3 by exploring different values of 1) sleeping ratio, 2) clients number, and 3) the ratio of shared data. The purpose of the experiment is to investigate how our scheme achieves the desired performance theoretically predicted including reducing the average query delay and the uplink request per query. We also made a comparison with AS and SACCS, we also consider the processing time of the base station. In our model the BS has a copy of all data items requested by users in its service area, and we are interested in investigating the effect of processing time in reducing the query delay, hence improving the whole performance of the network.

7.2 Experimental Results:

We simulated our proposed scheme of cache invalidation in a network basically composed of a BS and varying number of users.

7.2.1 Comparison at Different Sleeping Ratio:

Scenario 1 study the performance of AS, ASCCS, and ADBC with the sleep characteristic of the MU, Fig3.a shows a plot of the average query delay against varying sleeping ratio s, it is obvious that ADBC outperforms AS and SACCS. AS has the largest delay due to the fact that the user must connect to the server whenever a local miss happens, while in the SACCS and ADBC the data is being fetched from the base station or from the UIC respectively. In ASCCS, the base station has to take care of all queries, and serve them, however in crowded network it may not be able to attain serving all this number of users, thus the service will be slower, and the delay will accordingly be raised. On the other hand, substantial improvement in terms of average query delay is remarkably achieved using our scheme due to the distributing of the load from base station among all MUs, as a result the data will be delivered faster.

Fig 3.b expresses the relationship between miss-rate translated as uplink per query at different sleep rates. We observe proportional increase of the uplink request with increasing s, which can be explained as follow: as s increases, the MUs have less opportunity to capture all the invalidation reports, yet SACCS and AS have the same uplink request which is greater than ADBC since both of them after waking up must wait for the next query to validate their data, while in ADBC the mobile user asks for invalidation when he awakes. By reducing the average delay and uplink per query, we take care of two important issues in the system, achieving more fast service to the MU, and a good utilization of the available bandwidth.

7.2.2 Comparison at Different Number of Mobile Users:

Scenario 2 investigates the performance over different number of MUs, Fig 4, shows the delay time under different number of users, as can be noted, with increasing MUs, in AS approach an approximate linear increase in delay is depicted, while in both SACCS and ADBC the delay is predicted to be constant for high number of users. In our algorithm, increasing the number of users implies a reduction in the probability of miss rate at the BS cache, in view of the fact that more MUs mean more shared data are found in BS or UIC hence increasing the opportunity to serve the query without referring to the original data base. On the other hand, the uplink per query is constant along the variation of mobile users, in all the tested models; however our model still scores the least uplink query among the others.

7.2.3 Comparison at Different Shared Data Ratio:

Scenario3 study the performance over different shared data ratio, Fig 5 shows the average delay for the three scheme, In AS scheme each MU has its own HLC at the base station thus the MU is not affected by the shared data, this explains the stability in its average delay in the figure. As the ratio of shared data increases both SACCS and ADBC have a drop in the average delay due to the fact that as more data is shared, it is much likely for the MU to find the data at a nearer place either in the base station or at neighbors in both SACCS, and ADBC respectively, therefore getting it earlier instead of connecting to the server. Although our algorithm strongly depends on the shared data being high, this can be easy to take place nowadays as a consequence of the multiple services highly demanded.

7.2.4 Comparison at Different Query Rate:

Scenario 4 studies the performance over different query rate ratios. Fig 6.a and 6.b measure the average delay and uplink per query respectively as function of query rate at the three schemes. We can see the increase in both the number of uplinks and the average delay when increasing the queries per second, however the growing trend is varies among AS, ASCCS and ADBC with our proposed scheme achieving the lowest delay.

The reason behind this behavior depicted in both figures is that the query rate is an indicator of the frequency of accessing the data. Hence the data will be available at the cache of the MU and the query will be served locally resulting in a remarkable decrease in the delay, as well as uplink query. Even though the data may not exist at the MUC, there is a high probability that other users have a copy of the requested data, sparing the user from connecting to the server, thus reducing both delay and uplink query.

8-CONCLUSION

IR-based cache invalidation techniques have received considerable attention due to their scalability in spite of the problems associated with it such as long query delay and low bandwidth utilization. In this paper we focused on these problems and proposed a new simple and efficient cache invalidation scheme which exploits the existence of shared data among mobile users in distributing the control of the data via the interested users. By doing this, the heavy load on the base station would be reduced allowing the BS to handle other tasks thus enhancing system efficiency. In the new scheme ADBC, the BS delegates the control of this data to the MU, names him UIC and redirects any user who asks for this data to the corresponding UIC. Also, our scheme handles the problem of roaming by allowing the MU to send the ID of the new BS to the old one, enabling both stations to benefit from the valid data available at the MUC. In addition we have demonstrated analytically via simulation that ADBC can greatly improve the query delay and the uplink per query which means faster service for the MU, and a better utilization of the available resources. Our results confirmed our assumptions that in comparison with ASCCS and AS, the proposed ADBC a chives superior results in terms of query delay as well as uplink per query.

REFERENCES

- Barbara, D. and lmielinski, T. (1997) Sleepers and Workaholics: Caching Strategies for Mobile Environments, Proc. ACM SIGMOD, pp. 1-12.
- Cao, G. (2002) On Improving the Performance of Cache Invalidation in Mobile Environment, ACM Mobile Networks and Applications, vol. 7, no. 4, pp. 291–303.
- Cao, G. (2003) A Scalable Low-Latency Cache Invalidation Strategy for Mobile Environments, IEEE Trans. Knowledge and Data Engineering, vol. 15, no. 5, pp. 1–15.
- Hu, Q. and Lee, D. (1997) Adaptive Cache Invalidation Methods in Mobile Environments, Proc. High Performance Distributed Computing, pp. 264–273.
- Jing, J. , Elmagarmid, A. , Helal, A. S. and Alonso R., (1997) Bit-sequences: An Adaptive Cache Invalidation Method in Mobile Client/Server Environments, Mobile Networks and Applications, vol. 2, no. 2, pp. 115–127.
- Kahol, A., Khurana, S., Gupta, S. and Srimani, P. (2001) A Strategy to Manage Cache Consistency in a Disconnected Distributed Environment, IEEE Transactions on parallel and distributed systems, Vol. 12, No. 7, pp. 686-700.

- Vora, A., Tari, Z., Bertok, P. and Lai, K. (2002) A Mobile Cache Consistency Protocol Using Shareable Read/Write Time Locks, The 8th International Conference on Parallel and Distributed Systems (ICPADS).
- Wang, Z., Das, S., Che, H., and Kumar, M. (2004) A Scalable Asynchronous Cache Consistency Scheme for Mobile Environments, IEEE Transactions on Parallel and Distributed Systems, Vol. 15.
- Wang, Z., Kumar, M., Das, S., and Shen, H. (2006) Dynamic Cache Consistency Schemes for Wireless Cellular Networks, IEEE Transactions on wireless communication, Vol. 5, No. 1.
- Wu, K., Yu, P., and Chen, M. (1996) Energy-Efficient Caching for Wireless Mobile Computing, Proc. 20th Int'l Conf. on Data Eng., pp. 336-345.
- Yeung, M., and Kwok, Y. (2004) New Invalidation Algorithms for Wireless Data Caching with Downlink Traffic and Link Adaptation, Parallel and Distributed Processing Symposium.
- Yuen, K. (2000) Cache invalidation scheme for mobile computing systems with real-time data, Proc. ACM SIGMOD conf.
- Yuen, K., Tari, Z., and Bertok, P. (2006) an analytical study of cache invalidation algorithms in mobile environments, International Journal of Pervasive Computing and Communications, Vol 2.

تحقيق التوافقية في شبكات الخادم/الزبون النقالة باستخدام الذاكرة الوسيطة الموزعة

د. حاتم حماد، أستاذ مشارك، الجامعة الإسلامية.
م. حنان أبوثريا، هنسة كمبيوتر، الجامعة الإسلامية.
م. هدى حبوب، هنسة كمبيوتر، الجامعة الإسلامية.

لعل من أهم النظريات العملية لتحسين الأداء في الشبكات النقالة هو حفظ البيانات الأكثر استخداماً في الذاكرة الوسيطة للمستخدم، و مع ذلك تعتبر إدارة هذه البيانات وضمان توافقها مع البيانات المتوفرة في قاعدة البيانات عملية معقدة بالإضافة لخضوعها لكثير من التقييدات.وقد تم سابقاً إعداد عدة دراسات لمحاولة التغلب على هذه التقييدات ومن أهمها نظريا تقارير الإبطال، ورغم أنها حققت بعض النتائج المرضية إلا أنها تسبب التأخير في تنفيذ الاستعلامات حيث يضطر المستخدم للانتظار لضمان سلامة وتوافقية البيانات المتوفرة لديه قبل إجابة الاستعلامات.

في هذا البحث قمنا باقتراح نظام جديد معتمد على تقارير الإبطال، ولكنه يقلل التأخير الناتج عنها من خلال الإستفادة من البيانات المشتركة التي يتبادلها المستخدمون وكذلك عن طريق وضع ذاكرة وسيطة موزعة عند كل مستخدم للمشاركة في إدارة البيانات وتحمل جزء من العبء بدلاً من اقتصاره على المحطة المركزية.

ولتقييم النظام المقترح، تم عمل نموذج لبيان أنه يحقق نتائج أفضل من سابقيه على صعيد التأخير الناتج عن تنفيذ الاستعلامات.

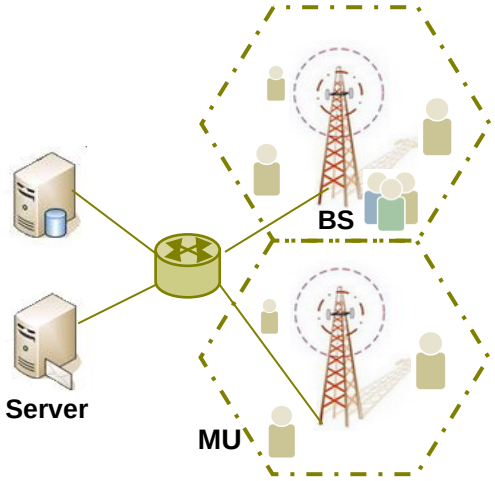

Fig.1 Wireless mobile environment**.**

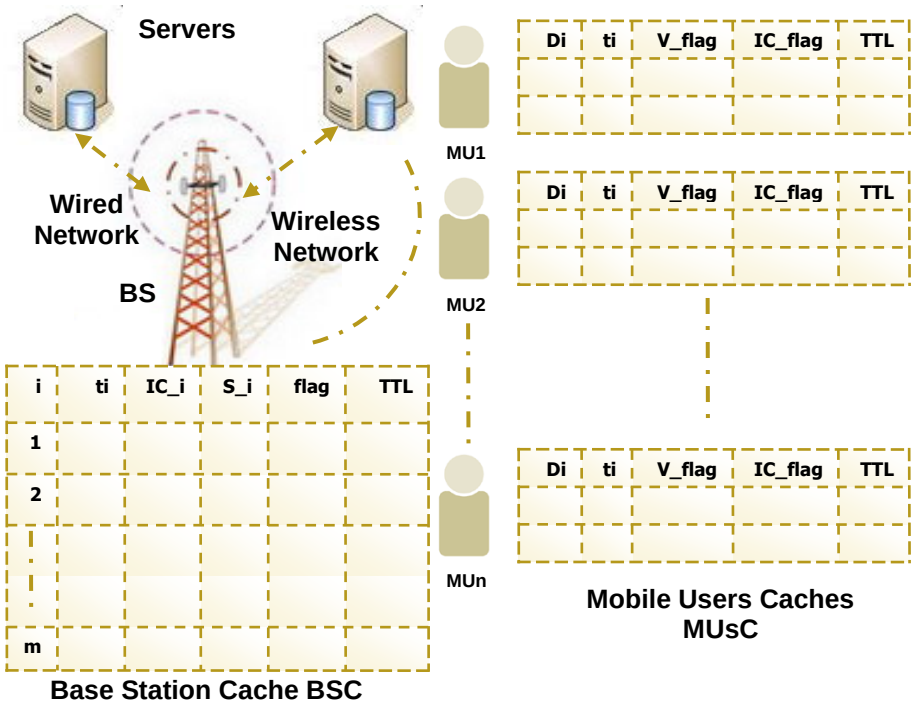

Fig.2 System Architecture Including Data Structure at Mobile User and Base Station.

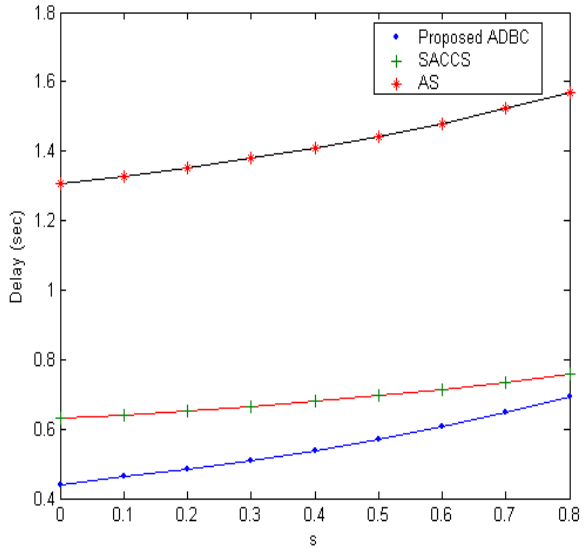

Fig.3a Average query delay against sleeping ratio

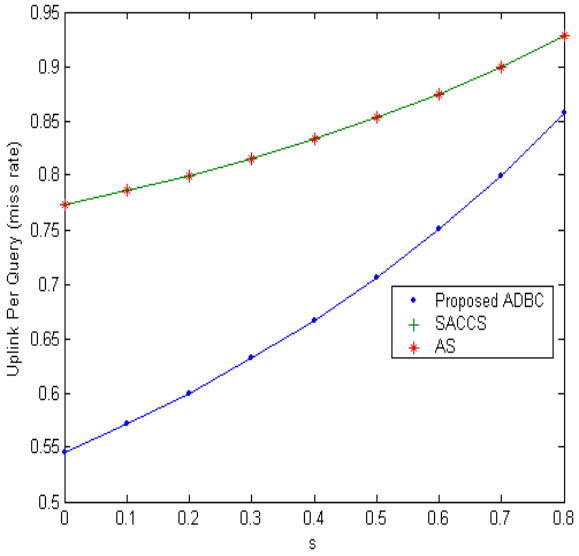

Fig.3b Uplink per query against sleeping ratio

3

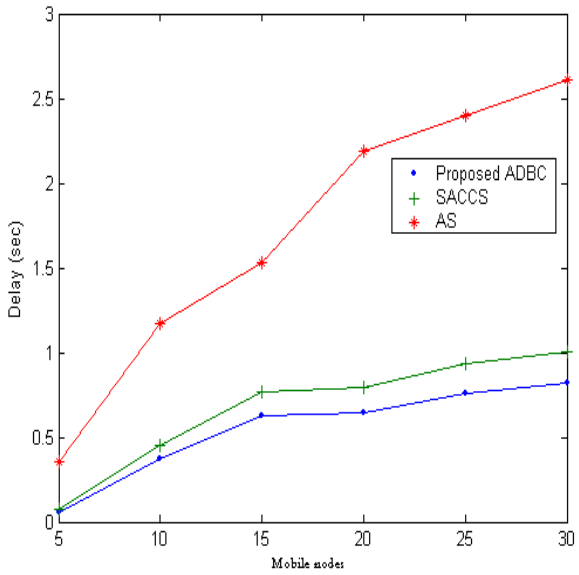

Fig.4 Query delay under variable users number.

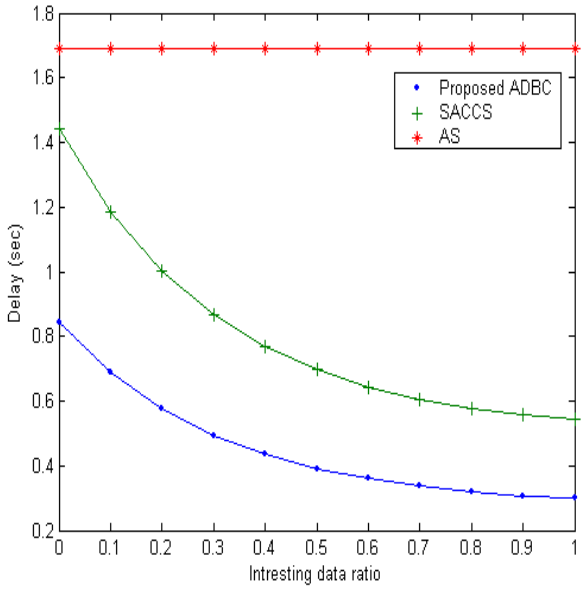

Fig.5 Query delay under variable shared ratio.

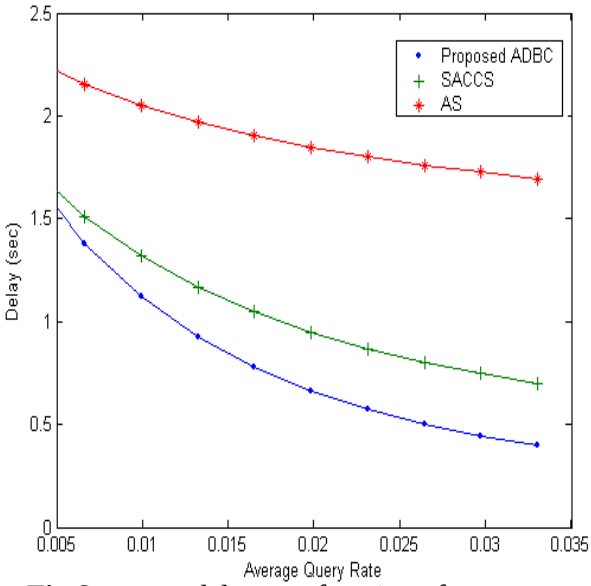

Fig.6.a query delay as a function of query rate

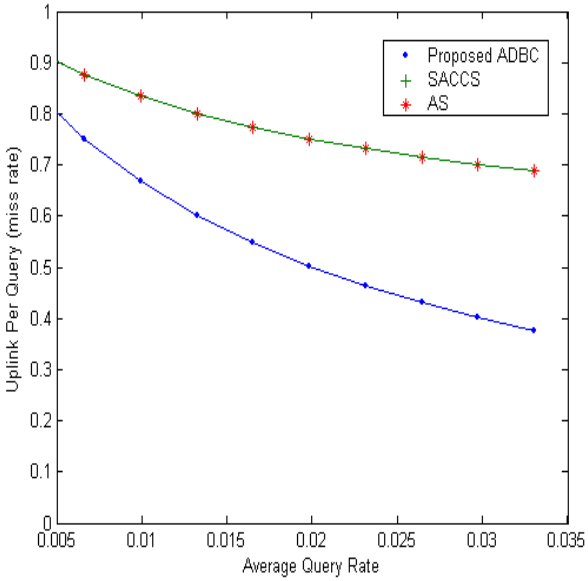

Fig.6.b Uplink per query as a function of query rate

Table 1: Data Structure of BSC and MUC.

Data structure maintained for every entry in BSC		Data structure maintained for every entry in MUC	
i	Identifier for data item cached by at least one MU in its coverage area.	***i***	Identifier for data item cached by at least one MU in its coverage area.
T_i	Time stamp of the last IR or data received from the server (the latest event).	***Di***	Data object for identifier i.
IC_i	The ID of Mobile user in charge, it is set to true when the user asks for data item i.	***Ti***	Time stamp of the last invalidation report or data received from the BS.
S_i	The ID of successor, particularly the BS toggles the old UIC to be the successor.	***V-flag***	Indicator for data validity, if the MU receives the IR, he sets the corresponding flag to zero.
Flag	A flag set false to assign the inconsistency of data at IUC,	***IC-flag***	The indicator that this Mobile user in charge related to that data.
TTL	Time to live, when it reaches zero, data is no longer valid	***TTL***	Time to live, when it reaches zero, the data is no longer valid
NBS	An identifier of the new BS visited by the UIC.		

Table 2: Estimation of Query Delay at at Different Circumstances.

No of users	**Only one MU request data**		**All MU's Request data**		**Average Delay**	
	D_{MU-DB}	D_{MU-BS}	D_{MU-DB}	D_{MU-BS}	D_{MU-DB}	D_{MU-BS}
5	0.01233	0.01226	0.07090	0.05453	0.03339	0.04162
10	0.01226	0.01225	0.26241	0.16070	0.08648	0.13733
15	0.01227	0.01221	0.32592	0.16727	0.08974	0.16910
20	0.01228	0.01226	0.51024	0.33354	0.17290	0.26126
25	0.01230	0.01226	0.55018	0.35381	0.18304	0.28123
30	0.01232	0.01232	0.59872	0.39641	0.20437	0.30553

Table 3 System Parameters and their Values.

tem	**Parameter Description**	**Value**
M	Number of data objects in the system	1000
N	Number of mobile nodes	5 ~ 30
λ_g	Average arrival rate of request query	1/120 query/s
μ_d	Average update of data object	10^{-4} update/s
s	Ratio of sleep time to sleep-awake period	0 ~ 0.8
d	Percentage of interested data	0 ~1.
C	Channel bandwidth	10Kbps