Performance Analysis of Improved Cache Invalidation Scheme in Mobile Computing Environment

Miraclin Joyce Pamila J.C. † and Thanushkodi K ††

†Senior Lecturer, Dept. of CSE, Government College of Technology, Coimbatore, India
†† Principal, Akshaya College of Engineering and Technology, Coimbatore, India.

Summary
Improved data processing capabilities and increased data storage capacity of mobile devices facilitate on-line internet transaction processing. When number of mobile clients involve in transaction processing, they make frequent requests for accessing the data item from the data base server. So such simultaneous access requests increase network overhead and thus poor response time. So to improve response time and to decrease network overhead, frequently accessed data items are cached. But data items that are cached should be properly invalidated and in this paper, an improved cache invalidation technique is proposed. The proposed technique provides implicit cache invalidation, increased transaction throughput, dependency preservation and decreased response time with reduced complexity in mobile devices. Performance analysis ensures decreased response time, increased throughput with reduced link cost.

Keywords:
Cache invalidation, Transaction processing in mobile networks, Consistency preservation, concurrency control.

1. Introduction

The increasing popularity of mobile devices has led to many commercial transactions to be executed from the mobile devices, thus enabling business on the move. Commercial transactions from mobile devices are significantly expensive as they involve computational and storage complexity.

Normally, M-Commerce applications have concurrent users to access the database items. Mobile computing environment enables multiple users to perform on-line transaction processing independent of their physical location.

Internet access from mobile devices is very expensive as limited bandwidth is available on wireless links that are unreliable. So when more number of Mobile Clients (MC) involve in database transaction with a web portal from a mobile device, the data access from the server is delayed, increasing the response time thus leading to decrease in transaction throughput.

So caching of frequently accessed items is needed, which will increase the availability of data and transaction throughput with reduced response time and network overhead.

Manuscript received September 5, 2009
Manuscript revised September 20, 2009

The remainder of the paper is organized as follows: Section 2 focuses on various existing cache invalidation techniques Section 3 discusses about the proposed reference architecture. Section 4 explains the improved cache invalidation technique while Section 5 analyses the performance. Section 6 concludes the presentation.

2. Cache Invalidation schemes

Frequently needed data items in the database server are cached to improve transaction throughput. The data in the cache must be properly invalidated, to ensure consistency of the data. Normally the data base server involves in cache invalidation, by sending Invalidation report (IR) to all the mobile clients. Effective cache invalidation strategies must be developed to ensure the consistency between the cached data in the mobile clients and the original data stored in the database server.
Generally, there are three basic ways to design invalidation strategies:
1. Invalidation with Stateful Server: The server knows which data are cached by which mobile clients. Whenever a data item is changed, the server will send an invalidation message to those clients which cached that particular item. This method necessitates the server to locate the clients. Since disconnected mobile clients cannot be contacted by the server, the disconnection of a mobile client automatically assumes that its cache is no longer valid upon reconnection. Also the mobile client needs to notify the server of its relocation. The mobility, disconnection of the clients and updation of data items will increase uplink and downlink messages.
2. *Validation of cache data by mobile client*: The clients that have cached the data items normally query the server to verify the validity of their caches, whenever any cached data is used or on reconnection after disconnection if any. This method generates lot of uplink traffic in the network.
3. *Invalidation with stateless Server:* The server is not aware of the state of the client's cache. The server simply periodically broadcasts an invalidation report containing the data items that have been updated recently. The client assures the validity of the data item by listening to the

report, going uplink only if the cache validity is no longer guaranteed.

Among all cache invalidation techniques, invalidation with stateless server is found to be suitable as it reduces uplink traffic alleviating the problem of locating the mobile client. The following section deals with various cache invalidation techniques with stateless server.

2.1. No-Checking Caching Strategy

The no-checking scheme proposed in [4][5] is the broadcasting timestamp (TS) technique. The Invalidation Report (IR) consists of the current timestamp T and a list of (o_i, t_i) such that $t_i > (T - w \times L)$, where o_i is object id, t_i is the corresponding most recent update timestamp and w is the invalidation broadcast window. The IR contains the update history of the past w broadcast intervals. When clients operate in energy efficient doze mode, they may miss some invalidation reports. The timestamp of the latest invalidation report received by the client (T_{lb}) helps them to decide whether the reports they missed are all included in the current window w or not. In the no-checking scheme, on disconnection of mobile client, the entire cache contents are discarded upon reconnection which significantly increases the bandwidth requirement. The algorithm for the same is given in Figure 1.

```
if (Tlb < (Ti - L x w))
        Drop the entire cache
else
        for every item oj in the cache
        if (oj ϵ IR) and (tjc < tj)
                Throw oj out of the cache
        else tjc = Ti
for ¥oi ϵ Qi
    if oi, is in the cache
        use the cache's value to answer the query
   else go uplink with the query
        Tlb = Ti
```

Figure 1. *TS* Client Cache Invalidation Algorithm

2.2 Bit- Sequence Algorithm

In *TS* algorithm, window size w is fixed. If the client is connected most of the time, a small broadcast window w is sufficient. But if disconnection period of mobile client is more than broadcast window size, the entire cache invalidated.

In the bit-sequence approach [8], the IR consists of a set of bit sequences, each of which has a corresponding timestamp. Each bit represents a data item in the database with a "1" bit in a sequence to indicate update of the corresponding data item and a "0" bit to indicate that the data item is not updated since that time. The set of sequences is further organized in B_n structure having N bits, which correspond to N data items in the database. The timestamp of the sequence B_n, denoted by TS (B_n), indicates the time after which these N/2 items have been updated. Generally, bit sequence B (k = 1, 2… n, 2^n = N) in the structure consists of $|B_k| = N/2^{n-k}$ bits of which as many as $N/2^{n-k+1}$ data items have been updated after the timestamp TS (B_k). A dummy sequence B_o is used, where TS (B_o) to indicate the time after which no data item has been updated. The client algorithm for BS strategy is given in figure 2.

```
if TS(Bo) ≤ Tlb
        Entire cache is assumed to be valid
if Tlb < TS(Bn)
        the entire cache is invalidated and stop.
Locate bit sequence Bj th timestamp such that
     TS(Bj) ≤ Tlb ≤ TS(Bj-1) ¥j (1≤j≤n)
        Invalidate all the data items represented by "1"
        bits in Bj .
```

Figure 2. BS Client Cache Invalidation Algorithm

2.3 Adaptive Invalidation Technique

To avoid bandwidth wastage of the downlink broadcast channels as in BS approach and uplink bandwidth wastage as in TS approach, two methods are proposed [9] that dynamically adjust IR according to system workload. They are Adaptive IR with Fixed Window (AFW) and Adaptive IR with adjusting window (AAW).

2.3.1 Adaptive IR with Fixed Window

IR consists of two types of information: Update history for window IR (w) with size $n_w(\log(N) + b\gamma)$ and Bit sequences structure IR(BS) with size $2N + b\gamma\log(N)$ where n_w is the number of data items updated within w broadcast periods, $b\gamma$ is the size of timestamp, and N is the number of data items in the database.

The server begins to broadcast the invalidation report periodically at time Ti = iL**.** The server keeps a list $IR(w_i)$ and a set of bit sequences IR(BS), where $IR(w_i)=\{(o_j,t_j)|\ o_j \epsilon D$ and $T_i - w \times L \le t_j \le T_i$ and IR(BS) = $\{B_n, TS(B_n), \ldots B_o, TS(B_o)\}$. Queries from long disconnected clients which reconnect during the intervals between T_{i-1} and T_i can be processed as shown in the algorithm given in a figure 3.

This method guarantees that *BS* is broadcast as the next invalidation report only if there is at least one client which needs more update history information than the window *w* can provide. Otherwise, window *w* is broadcast whenever it is possible to save the bandwidth of the broadcast channel.

```
Server:
if ( i ϵ C and TS(Bn) ≤ Tlb i < T - w x L)
  Server selects IR (BS) as the next IR
else
  Server broadcasts IR (wi) as the IR.

Client:
if (report type is IR (BS))
    Run BS client cache invalidation algorithm
else if (Tlb < Ti - w x L and not yet sent Tlb to server)
        Send Tlb to the server
     else
        Invalidate its cache with TS algorithm
```

Figure 3. AFW Cache Invalidation Algorithm

2.3.2 Adaptive IR with Adjusting Window

When disconnection of mobile clients that is barely longer than the window w, the use of *BS* invalidation reports is still quite wasteful. It is sufficient that the window size w of the *TS* algorithm is adjusted a little bit, so that cache invalidation is made possible with TS report that considerably saves the downlink cost for invalidation reports. For very long disconnection, adjusting the window size alone may result in a large number of update information and consequently large invalidation reports to be broadcast. Thus, the *BS* report should be a better choice. Integrating these three types of reports (i.e., default TS**,** varying window size, and *BS)* yields another adaptive cache invalidation algorithm [9], adaptive with adjusting window *(AAW).*

The approach in [5] adjusts the window size for each data item according to changes in update rates and reference frequencies of the item. It requires uplink communication for the server to obtain feedback information about the access patterns from the clients. Further, the window size w must be contained in each report for the client to be able to correctly invalidate its cache. The algorithm for AAW is given in Figure 4.

2.4 Invalidation with AVI

Huen et. al [11] propose a cache invalidation scheme named as Invalidation by Absolute Validity Interval (IAVI). An absolute validate interval AVI, for each data item based on its real-time property is defined. A mobile client can verify the validity of a cached item by comparing the last update time and its AVI. A cached item is invalidated if the current time is greater than the last update time by its AVI. With this self-invalidation mechanism, the IAVI scheme uses the invalidation report to inform the mobile clients about the change of AVI rather than the update event of the data item.

The Absolute Validity Interval of a data item can be derived based on the previous update intervals of the data item. It is assumed that each update is associated with a time-stamp and after the completion of the update, the time-stamp will be recorded with the updated data item. The time-stamp together with the AVI of the data item can be used to determine its validity. The update intervals can be used to define the validity periods of a cached item and it is said to be valid until the time of the next update.

```
Server:
if ( i ϵ C and TS(Bn) ≤ Tlb i < T - w x L)
   if(sizeof IR(BS) > sizeof IR(w'))
      Server selects IR(w') as the next IR
   else
     Server selects IR(BS) as the IR.
else
    Server broadcasts IR(w) as next IR

Clients:
 if (report type is IR(BS))
      Run BS client cache invalidation algorithm
 else if (Tlb < Ti - w x L and  Tlb< t with dummy id as its
       id and not yet sent Tlb to server)
        Send Tlb to the server
     else
        Invalidate its cache with TS algorithm
```

Figure 4. AAW Cache Invalidation algorithm

The server algorithm consists of two parts, invalidation report generation and AVI adjustment. The former deals with the selection of the information to be included in the invalidation report. Since the broadcast bandwidth is a valuable resource that is shared among large number of mobile clients, it must be used efficiently. AVI adjustment is concerned with adapting the AVI values of the data items to achieve the desired level of the cache coherence for the system. In order to notify the mobile clients about the changes in AVI values, invalidation reports are generated and disseminated periodically. The invalidation report contains a data ID and its update time, whose update interval must satisfy the expression:
$T_{update(i,\ n)} - T_{update(i,n-1)} < AVI_i\ (l - F_i)$, where i refers to data item, $T_{update(i,n)}$ is the timestamp of n^{th} update, AVI is the valid life span of data item i, F_i is the AV1 tolerance for data item i.

Cache invalidation on client side is divided into two parts, implicit invalidation and explicit invalidation. In implicit invalidation, a cached item is invalidated by the expiration of its AVI. This occurs when the cached item is accessed and the AVI is found to have expired. In contrast, invalidation report is from the server.

a) Implicit Invalidation**.** When a cached item is referenced by a transaction from the mobile client, the transaction will examine the update time and AVI value of the item. If the sum of the update time and the AVI value is smaller than

the current time, the item is assumed to be invalid. By using implicit invalidation, traffic cost for invalidation is minimized since a client can invalidate its cached items without waiting for additional information from the server. Moreover, when the client reconnects to the mobile network after disconnection, it can invalidate its cached copy without waiting for the next invalidation report from mobile server. If the AVI has expired for a cached item, it can be invalidated without extra verification. However, that if the cached copy seems to be valid based on its AVI, the client needs to reference the first invalidation after reconnection report to determine the validity of its cached items.

b) Explicit Invalidation**.** In the explicit invalidation scheme, the size of invalidation report is much smaller than other techniques such as Bit Sequence (BS) and timestamp (TS). Unlike BS and TS, which include every update event in invalidation reports within certain period or interval, the invalidation report of the AVI scheme only contains the entries for the data items whose AVIs have been modified . Moreover, if the AVI and the update interval of the data items are reasonably well matched, many of the update events do not generate explicit invalidation reports.

3. Proposed Reference Architecture

The Figure 5 shows the reference architecture. It consists of two major components, Mobile Client (MC) and a Fixed Network.

Transactions are initiated and terminated at a mobile host itself. A host that can move while retaining its network connection is a mobile Client. A static network consists of fixed hosts and Base stations (BS).

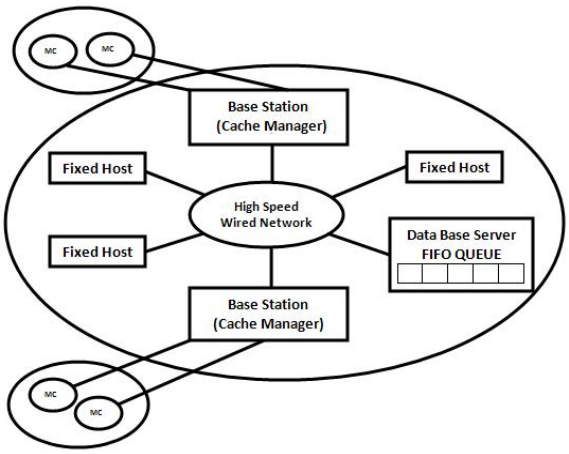

Figure 5 Reference Architecture

Mobile clients may not always be connected to the fixed network. It may be disconnected to save battery consumption. Hence disconnections are handled as normal situations and not as failures. Mobile host may differ with respect to the computing power and storage space; however MC can run a DBMS module.

BS that interacts with MH and wired network acts as a gateway between wired and wireless networks. As the BS, the fixed host acts as a gate way, all the requests made by any mobile client to the data base server will be forwarded by the respective BS. So the base station is assumed to keep track of the requests made by any mobile client.

In the proposed architecture, the base station, a fixed host is found be suitable for storing a cache. A cache manager (CM) module in it monitors and coordinates the operations in a cache. If any fixed host accesses the database server it may also have a local cache with cache manager having the same functions defined. A database server holds the data that can be accessed by any mobile client and fixed host. It normally resolves the update requests in FIFO ordering.

4. Proposed Cache Invalidation Technique

The proposed technique assumes that the fixed hosts i.e. base stations that serve the mobile hosts may have cache stored in it instead of caches in mobile clients. The cache manager module is loaded in the fixed host and it assures controlled concurrent access to the cached data. It monitors validity of cached data and sends invalidation indication when data is updated. It also ensures recovery of affiliated transactions.

The cache manager module initially fetches the data item from the database server and stores it in the cache. All subsequent requests for the data item will be serviced by the cache manager itself. This reduces network overhead. The cache manager stores the data as quintuple in the cache which includes unique-Id of the data item, Last updated Time, Predicted Life Period, i.e. predicted time period in which the data item value is valid, current value of the data item, details of affiliated transactions, i.e. transactions that simultaneously access the data item.

In the proposed Predicted Life Period (PLP) method, the valid life span of data is predicted based on the probability of updation of the data item. Within PLP interval, the data item is assumed to be valid and all the mobile clients can access the data item within PLP and thus increase in degree of concurrency is achieved. But at the same time when there is an updation request from one of the clients is received at the cache or when PLP of any data item expires, it is immediately invalidated with change in PLP. This ensures consistency preservation.

Also since the cache is maintained in the fixed host and PLP associated with data item is in the cache, independent execution in multiple mobile clients with an assumption of valid data is much reduced. Because in

previous approaches [5][7][11], the data item is copied locally to all mobile hosts and they initiate the transaction execution independently. So a definite increase in abortion of almost all transactions except the one that commits first.

Since the predicted probability is based on frequency of updation, the value of PLP is dynamically changed reflecting the data access environment. The significance of the PLP value is that, if there is no frequent updation of the data, the time period in which data is valid is very high. It need not be invalidated unnecessarily with limited PLP. This avoids unnecessary periodic invalidation report. Conversely if the updation frequency of data item is very high the PLP is reduced to ensure frequent invalidation.

The proposed method generates Invalidation Report only when the data item is changed against periodic invalidation in previous approaches. And also since the invalidation report is generated only for the updated data item the size of the IR is almost fixed assuming fixed data item size. So the down link capacity requirement is much reduced.

4.1 Algorithm

Cache Manager:
if ($T_{current}$ is within PLP and data in cache)
  Allow data item access from cache and update Predicted Life Period
else
  Fetch data from the server
if (update request)
  Update locally and immediately send invalidation indication to all affiliated transactions.
  Forward update request to server.

Server:
If (data access request)
    Broadcast data item to cache
if (update request)
    Update the data item in the server and send invalidation confirmation.

The proposed technique supports immediate dissemination of invalidation indication to all fixed and mobile hosts that have accessed the data items earlier in the event of updation. So all the transactions are forced to refresh the value of the data item. Simultaneously the threaded cache manager forwards the update request made by the mobile client to the server. The server after validation does the updation and broadcasts the invalidation confirmation together with updated value. On receiving this invalidation confirmation the cache manager updates the quintuple in the cache. If the data item is not updated within PLP, then cache manager recalculates PLP when it expires.

Since cache is maintained in the fixed host, the complexity in the mobile client is very much reduced against the technique in which mobile clients store the data item locally.

5. Performance Analysis

5.1 Comparison of cache invalidation Protocols

The various existing invalidation protocols are compared with proposed invalidation scheme against various parameters and the comparison results are tabulated below.

Parameters	TS	BS	AFW	AAW	IAVI	Proposed
Location of cache	Mobile Client	Mobile Client	Mobile Client	Mobile Client	Mobile Client	Fixed Host
Invalidation Report	Periodic	Periodic	Periodic	Periodic	Periodic/ On need	On need
Probability of Abortions	High	High	High	High	Moderate	Low
Complexity in MC	High	High	High	High	Moderate	Low
Information Dissemination	Broadcast	Broadcast	Broadcast	Broadcast	On Demand/ Broadcast	On Demand / Broadcast
IR Generation	DB Server	DB Server	DB Server	DB Server	DB Server	Fixed Host
IR Size	$n*(b\gamma+I_{Size})$	$(B_k*\gamma)*(N/2^{n-k})$	$n_w(\log N+b\gamma) / 2N+b\gamma\log N$	$\min(n_w(\log N+b\gamma), 2N+b\gamma\log N)$	$I_{Size}+b\gamma$	$I_{Size}+b\gamma$
Complexity in the Server	High	High	High	High	High	Low

Table 1. Performance comparison of all Cache Invalidation Techniques

5.1.1. Complexity in Mobile Client

In all the previous algorithms discussed above, the cache is maintained in the mobile client. Though the mobile devices are equipped with much storage and processing capabilities, it is desirable to have reduced complexity involved in executing invalidation algorithms in mobile client. The proposed approach assumes that the cache is maintained in the fixed host that serves as a base station. So the complexity involved in cache storage or cache invalidation in mobile client is alleviated. Also storing cache in fixed host rather than in mobile client ensures reduced instances of replicated data, which is equal to number of base stations against number of mobile clients in previous approaches. If large number of mobile clients involves application processing, the proposed PLP technique barely reduces the replicated instances which assure improved consistency preservation.

5.1.2 Frequency of invalidation report

In all previous approaches the invalidation report is broadcasted by the server periodically. Whereas in the proposed technique, the invalidation report is generated only when the server finds that the data item set is updated reducing unwanted periodic invalidation. So considerable reduction in the down link capacity is achieved. In the proposed technique, Invalidation Confirmation alone is broadcasted to all cache holders and that too only on invalidation indication from one of the cache holders indicating possibility of updation. The invalidation reports are communicated between only fixed hosts (cache holders) in the wired network which very much ensures effective utilization of wireless channels.

5.1.3 Probability of abortions

In the proposed approach, since the cache is stored in fixed host rather than mobile client, the affiliated mobile clients are immediately informed by the fixed host when there is an invalid data as a result of updation. In the previous approaches since cache is stored in the mobile client all affiliated mobile clients continue with their execution even with invalidated data item, since explicit cache invalidation is sent periodically. But in the proposed technique, the invalidation indication is immediately broadcasted. Though this broadcast of invalidation indication acquires bandwidth, this helps in drastic reduction in number of mobile clients getting aborted. And also the broadcast is on need and not periodical.

5.1.4 Transaction Throughput

The proposed technique stores the cache in fixed host and assures the predicted life period of the data item to be very close to the actual life period as PLP is based on update pattern. This allows more transaction to continue with their execution thus increase in transaction throughput. Almost all the techniques support increase in transaction throughput but as discussed earlier they result in increase in abort probability. Almost all other affiliated transactions are aborted when one of the affiliated transactions commits.

5.1.5 Size of the invalidation Report

The Invalidation Report size for BS, TS, AFW and AAW is proportional to the database size N whereas for IAVI and proposed scheme it is comparatively very less which is equal to sum of time stamp size and data item size. Because in IAVI and proposed technique the invalidation report is generated only on updation of cached data item , whereas in the previous techniques invalidation report consists of status of almost all data items that are cached in the database.

5.2 Performance metrics

The proposed Technique is simulated and performance metrics are analyzed. The following graphs show the results of simulation.

5.2.1 Response Time Vs Number of transactions

Response time refers to the time taken to service the request that is made from the mobile client. In figure 6, the response time analysis between direct database server access and cache access is given.

From the figure 6 it is understood that, as the number of mobile transactions increases, the response time increases for direct database server access. It is increased exponentially, as number of transactions increases that results in network overhead with increased load at the server. But the response time of the proposed approach where cache is maintained in BS is compariteVly low when compared to direct database server access.

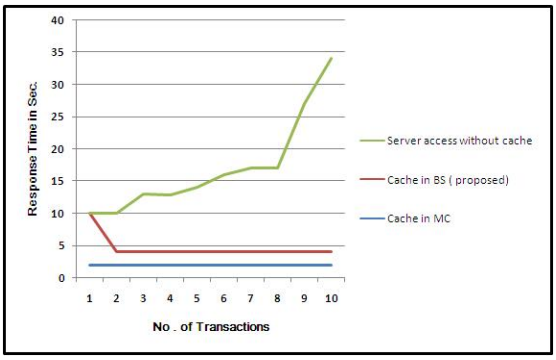

Figure 6. Response Time Vs No. of Transactions

The response time is very low and constant, when cache is in the mobile device itself. So BS,TS,AAW, AFW and IAVI techniques almost have fixed response time. But the complexity of handling invalidation report generation and storage constraints in the mobile client become dominant issues to be managed , when mobile client itself stores the cache.

So it is found that the response time is considerably good and decreased when cache is in fixed host . This also ensures reduced complexity in mobile clients.

5.2.2 Probability of aborting transactions Vs Number of Transactions

In figure 7 the probability for aborting transaction is analysed between techniques with cache in mobile client and proposed technique with cache in fixed host.

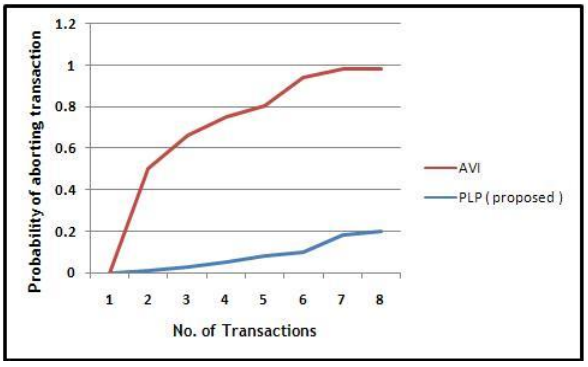

Figure 7 Analysis of probability to abort transactions

From the figure 7 it is understood that the probability of affiliated transactions getting aborted is very high in mobile client based invalidation approaches. It is because, between two consequtive periodic invalidation reports or within AVI interval all mobile clients assume that the data items are valid and they continue with local execution resulting in local updation of the data item. But when more number of mobile clients involve in transaction execution, only the moble client that first commits the execution is ultimately allowed to update the data item in the server resulting in abortion of all other transactions.

But in the proposed approach though all the mobile clients are allowed to access the data item simultaneously within PLP, when one of the clients requests for updation, the cache manager manages the request and sends invalidation indication to all other mobile clients. This forces all the clients to refresh the value of the data item resulting in reduced probability of aborting a transaction.

The probability rise in the proposed method is due to the delay in transmission of invalidation indication. This is increased as more number of mobile clients get involved in transaction processing. But the increase is comparatively low and acceptable.

5.2.3 Invalidation Report Size Vs Database Size

The size of the invalidation report determines the required downlink capacity. In figure 8 the downlink capacity of various techniques are compared and analyzed.

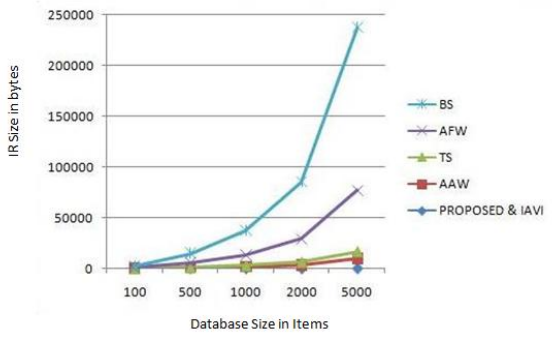

Figure 8 Invalidation Report Size Vs Database Size

In TS technique the invalidation report size is based on the updation rate of the cached data items within periodic interval. But in BS approach the invalidation report size is based on the number of database items and also the updation frequency. Thus the IR Size increases with increase in database size. In AFW method the invalidation report size is determined based on whether client needs update history or Bit sequence structure. So the IR Size varies between TS and BS techniques. In AAW since the window size is adapted to update frequency and so the invalidation report size is comparatively small. Both in IAVI and proposed PLP technique, the invalidation report is sent for every individual data item, so the R size is drastically reduced.

6. Conclusion

The proposed technique provides concurrency control without locking, with dynamic PLP of data item that increases the transaction throughput. The performance analysis ensures decreased response time, reduced complexity in mobile device with notable reduction in probability of aborting transactions and reduced downlink requirement.

References

[1] Ramez Elmasri, Shamkant B.Navethe, "Fundamentals of database systems", McGraw Hill, 2006.

[2] Victor C.S., Kwok wa Lam and Son, S.H., "Concurrency Control Using Timestamp Ordering in Broadcast Environments", The Computer Journal, Vol.45 No.4 PP.410-422, 2002.

[3] Ho-Jin Choi, Byeong-Soo Jeong, "A Timestamp-Based Optimistic Concurrency Control for Handling Mobile Transactions", ICCSA 2006, LNCS 3981, pp. 796-805, 2006.

[4] P. Krishna Reddy, Masaru Kitsuregawa, "Speculative Lock Management to Increase Concurrency in Mobile Environments", MDA'99, LNCS 1748, pp. 82-96, 1999.

[5] Salman Abdul Moiz, Dr. Lakshmi Rajamani, "Single Lock Manager Approach for Achieving Concurrency in Mobile Environments", Proceedings of 14th IEEE International Conference on High Performance Computing, 2007.

[6] Salman Abdul Moiz, Mohammed Khaja Nizamuddin," Concurrency Control without Locking in Mobile Environments", ICETET 2008.

[7] Barbara, D. and Imielinski, T., "Sleepers and Workaholics: Caching Strategies in Mobile Environments", Proceedings of ACM SIGMOD, 1994**.**

[8] Jing, J., Elmargarmid, A., Helal, A. and Alonso, F., "Bit-Sequences: an Adaptive Cache Invalidation Method in Mobile Client Server Environments", Technical Report CSD-TR-94-074.Purdue University, May 1995.

[9] Hu, Q. and Lee, D.L., "Adaptive Cache invalidation Methods in Mobile Environments", Proc. 6th IEEE International Symposium on High Performance Distributed Computing Environments, 1997.

[10] Joe Chun-Hung Yuen, Edward Chan, Kam-Yiu Lam, H. W. Leung, "An Adaptive AVI-based Cache Invalidation Scheme for Mobile Computing Systems", IEEE, 2000.

[11] Hien Nam Le , Mads Nygard, "A transaction frame work for mobile data sharing services", The second International conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies, IEEE, 2008.

[12] Dunham, M. H. et al. "A Mobile Transaction Model that Captures Both the Data and Movement Behavior. Mobile Networks and Applications", 2(2): 1997.

Miraclin Joyce Pamila J.C. is a senior Lecturer in Department of Computer Science and Engineering, Government College of Technology, Coimbatore, Tamilnadu, India. She received her Master degree in Computer Science and Engineering in the year 2005 from Anna University, Chennai, India. Her fields of interests are Mobile Computing, Database management Systems, Network Security and Recovery system design for mobile networks. She is a life member of ISTE. She also teaches and guides modules at both B.E. and M.E. levels in Computer Science and Information Technology. She has published 14 technical papers in national and international conferences and journals.

K. Thanushkodi is the Principal of Akshaya College of Engineering and Technology, Coimbatore, Tamilnadu, India. His research interests are in the area of Computer Modeling and simulation, Computer Networking, Network security and Power Systems and Design. He received the B.E. in Electrical and Electronics Engineering, M.Sc. (Engg) from Madras University, and PhD in Electrical and Electronics Engineering from Bharathiar University, Coimbatore in 1972, 1976 and 1991 respectively. He has published 31 technical papers in National and International Journals.