# A New Cost Function based Cache Replacement Policy for Location Dependent Data in Mobile Environment

Ajey Kumar, Manoj Misra and A. K. Sarje

***Abstract*—Caching frequently accessed data items on the client side is an effective technique to improve the system performance in mobile environment. Due to cache size limitations, the choice of cache replacement techniques used to find a suitable subset of items for eviction from cache becomes important. In this paper, we propose a new cost function based cache replacement policy, Predicted Region Based Replacement Policy (PRRP) for location dependent data in mobile environment. Unlike earlier cache replacement policies that consider only direction/non-directional data distance, PRRP takes into account data distance integrated with predicted region of client's movement that adapts to client's movement nature. The results of simulation experiments show that our algorithm significantly improves the performance in terms of cache hit ratio, compared to previous schemes.**

***Index Terms*— Cache replacement, location dependent data, mobile computing, valid scope.**

## I. INTRODUCTION

LOCATION Dependent Information Services (LDIS) provide users with the ability to access information related to their current location. The need for LDIS arises frequently. By including location as a part of user's context information, service carriers can provide many value-added applications targeted specifically at mobile users such as travel and tourist information services, assistant and emergency services, nearest object searching and local information access.

Users of LDIS face many difficult challenges inherent to mobile environments. These include limited bandwidth, client power and intermittent connectivity [1,2,4,6].Caching helps to address some of these challenges. Caching frequently accessed data item on client side is an effective technique to improve data accessibility and to reduce access cost. Due to the limitations of cache size on mobile devices, it is impossible to hold all the accessed data items in the cache. As a result, cache replacement algorithms are used to find suitable subset of data items for eviction from cache. An efficient cache replacement policy is vital to ensure good cache performance.

The Authors are with Department of Electronics and Computer Engineering, I.I.T. Roorkee, Roorkee, India.
e-mail: {ajeykdec,manojfec,sarjefec}@iitr.ernet.in.

Traditional cache replacement policies, such as LRU and LFU, rely on the temporal locality of client's access pattern. However, in mobile networks where clients utilize location dependent information services, clients access pattern do not only exhibit temporal locality, but also exhibit dependence on location of data, location of the client and direction of the client's movement [4,5].Relying solely on temporal locality when making cache replacement decisions will result in poor cache hit ratio in LDIS.

Most of the existing cache replacement policies use cost functions to incorporate different factors including access frequency, update rate, size of objects, etc., however very few of these policies account for the location and movement of mobile clients. Cache replacement policies such as LRU, LFU and LRU-k [11,13], only take into account the temporal characteristics of data access, while policies such as FAR [9] only deal with the location dependent aspects of cache management but neglect the temporal properties. Only policy which does consider both spatial and temporal properties of data objects is the PAID policy [5] and MARS[7] .However, unlikely PAID, MARS does take into account updates to data objects.

In this paper, we are predicting a square area in the vicinity of client's current position, and give priority to the nearby data items in cache irrespective of the client's movement direction. Then, we define a cost based cache replacement policy *Predicted Region Based Replacement Policy* (PRRP) which takes into consideration the access probability, valid scope area, data size in cache and data distance based on the predicted square region, which has not been considered in the existing policies.

In the performance evaluation, *cache hit ratio* is employed as the primary performance metric. C*ache hit ratio* can be defined as the ratio of number of queries answered by the client's cache to the total number of queries generated by the client. Specifically, the higher the cache hit ratio, the higher the local data availability, the less the uplink and downlink costs, and the less the battery consumption.

The rest of the paper is structured as follows: Section II, reviews the related work. Section III, describes mobile system model used in this paper. Section IV, details the new cost based replacement policy PRRP. Section V, describes the simulation model carried out in this paper. Section VI, deals with performance evaluation and comparison. Section VII concludes the paper.

## II. Related Work

Cache performance depends heavily on replacement algorithms, which dynamically select a suitable subset of objects for caching in a finite space. Developing such algorithms for wide area distributed environments is very challenging. Temporal-based cache replacement strategies, such as LRU (least recently used), LFU (least frequently used) and LRU-K [13] have been studied widely in the past. These polices are based on the assumption that client's access patterns exhibit temporal locality (i.e. objects that were queried frequently in the past will continue to be queried frequently in the future). As a result, they are unsuitable for supporting location dependent services because they do not take into account the location of data objects and the movement of mobile clients. To overcome this problem, several location-aware cache replacement policies have been proposed specifically for location dependent services. Here we describe the most representative ones.

The Manhattan Distance-based cache replacement policy [10] supports location dependent queries in urban environments. Cache replacement decisions are made based on the distance between a client's current location and the location of each cached data object. Objects with the highest Manhattan distance from the client's current location are evicted at cache replacement. While the Manhattan based policy accounts for the distance between clients and data objects, it is still limited because it ignores the temporal access locality of mobile clients and the direction of client movement when making cache replacement decisions.

The FAR policy (Furthest Away Replacement) [9] uses the current location and movement direction of mobile clients to make cache replacement decisions. Cached objects are grouped into two sets. Data objects in the out direction set are always evicted first before those in the indirection set. Objects in each set are evicted in the order based on their distance from the client. Similar to the Manhattan approach, FAR only deals with the impact of client location and client movement on cache performance and neglects the temporal properties of clients' access pattern. It is also ineffective when mobile clients change direction frequently as membership of objects will change frequently between the in-direction and the out-direction sets.

In PAID (Probability Area Inverse Distance) [5] the cost function takes into account the access probabilities ($P_i$) of data objects and area of their valid scopes $A(vs_i)$ and the distance between the client's current position and the valid scope of the object concerned (known as data distance). But, the effect of the size of data object and the priority for the nearby data objects in cache from client's current position have not been accounted. Mobility Aware Replacement Scheme (MARS) [7] is also a cost based policy which comprises of temporal score, spatial score and cost of retrieving an object. Unlike PAID, it takes into account the updates of data objects. But as far as location–dependent data (LDD) is concerned their update rate (if exist) is negligible as compared to temporal data. Thus, for LDD, only spatial score dominates which consists of area of valid scope, data distance from current client location and data distance from anticipated client location. The impact of anticipated region in deciding cache replacement still remains unexplored.

None of these existing cache replacement policies are suitable if client changes it direction of movement quite often. As they only consider the data distance (directional/undirectional) but not based on the predicted region/area where the client can be in near future.

## III. System Model

We assume a cellular mobile network similar to that in [1, 2] as the mobile computing infrastructure (Fig. 1). It consists of two distinct sets of entities: mobile clients (MC) and fixed hosts. Some of the fixed hosts, called mobile support stations (MSSs), are augmented with wireless interfaces. An MSS can communicate with the MCs within its radio coverage area, called a wireless cell. An MC can communicate with fixed host or server via an MSS over a wireless channel. The wireless channel is logically separated into two sub channels: an uplink channel and a downlink channel. The uplink channel is used by MCs to submit queries to the server via an MSS, while the downlink channel is used by MSSs to disseminate information or to forward the answers from the server to target client. A mobile client can move freely from one location to another while retaining its wireless connection. Seamless hand-off from one cell to another is assumed. The information system provides location dependent services to mobile clients. We refer to the geographical area covered by the information system as the service area. A data item can show different values when it is queried by clients at different locations. Note that data item value is different from data item, i.e., an item value for a data item is an instance of the item valid for a certain geographical region. For example, "nearest restaurant" is a data item, and the data values for this data item vary when it is queried from different locations.

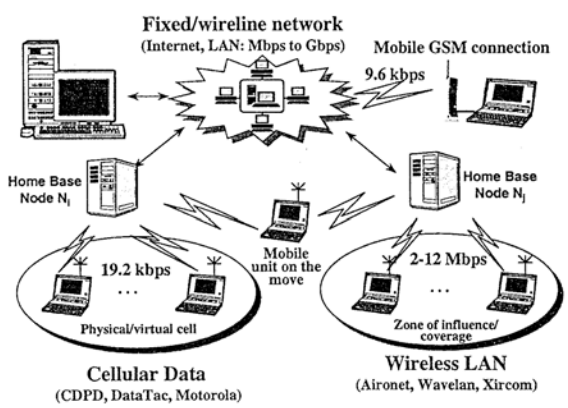


Fig. 1. Mobile Computing Infrastructure

In this paper, we assume a geometric location model, i.e., a location is specified as a two-dimensional coordinate. However, it can be easily extended to 3-dimension space by including the third dimension. Mobile clients can identify their locations using systems such as the (GPS) [3]. The valid scope of an item value is defined as the region within which the item value is valid. In a two-dimensional space, a valid scope *vs* can be represented by a geometric polygon $p\ (e_1, \ldots, e_n)$, where $e_i$'s are endpoints of the polygon. Data values are assumed to be read-only. A mobile client can cache data values on its local disk or in any storage system that survives power-off.

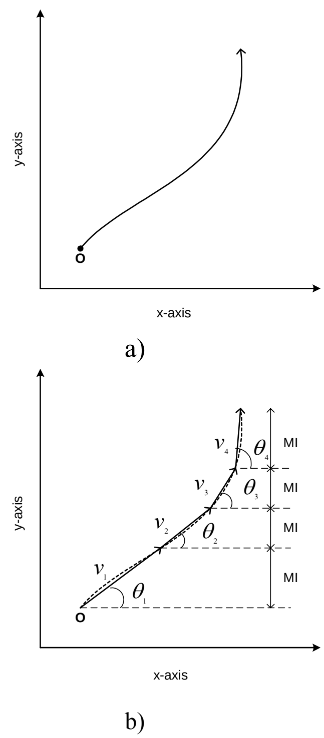


Fig. 2. Client's Movement Path a) Abstract Model b) Discrete Model

We also assume an unconstrained network, where mobile clients move freely inside the geographical region covered by the mobile network (without any restrictions). In abstract model the path of moving client is represented by a curve in 2-dimension, as shown in Fig. 2(a).Though abstract model is simple but from implementation point of view in computers discrete model is preferred [8].The path traveled in discrete model is modeled as a sequence of line segments, each associated with fixed velocity and direction, as shown in Fig. 2(b). Length of the line segment depends on change in direction and velocity. For random movement this duration between change in direction and velocity is small and for regular movement/highway users this duration is large. This duration is called as *Moving Interval* (MI) [5]. The distance between any two locations/points is the length of a direct line connecting the two points (i.e. Euclidean distance).

## IV. Predicted Region based Replacement Policy (PRRP)

Traditional cache replacement policies, due to their temporal nature, consider access probability as the most important factor that affects cache performance. A data item with least access probability is being replaced. LDIS for being spatial in nature needs that the distance of data item from client's current position and its valid scope area to be taken into account for cache replacement. In LDIS, the server responds to the user query with suitable data value with respect to client's current location. Greater the distance of valid scope of data value from the user's current position lower is the chance to become usable again since it will take some time before the client enters the valid scope area. Thus, it is better to eject the farthest data when replacement takes place. This reason is invalid when client continues to move away from current position. Because locations in the opposite direction of client's movement have very low chance of being revisited, even though they may be very close to it. Based on this reasoning the schemes like FAR,PAID(directional) assign higher priorities to data items in the client's direction of movement However, with random movement patterns of clients, the time it takes for the client to traverse a distance is not always directly proportional to the distance [5].Hence, it is useful to calculate the data distance with respect to the predicted region of user influence so that the data items in the vicinity of client's current position are not purged from cache. On, the other hand, larger the value of valid scope area of the data item, higher is the probability that the client requests this data. This is because, generally, the client has a higher chance of being in large region than small regions. Space require to store data items in cache also plays an important role in cache replacement. Keeping smaller size data items in cache helps to accommodate large number of data items in the cache. Hence, a promising cache replacement policy should select a victim data with a low access probability, a small valid scope area falling outside the predicted region and large data size.

In a MI, the user direction and velocity is known. But, one can't predict the same for the next MI (unless one knows the entire path of the user in advance like ships, trains, etc). We are not assuming predefined path of mobile user or predefined destination, so as to predict path towards the destination. We are considering a generalized scenario. So, instead of predicting path, we are more interested in predicting region. At the end of each MI, direction is selected randomly between 0° to 360° degrees and velocity between minimum speed $v_{\min}$ and maximum speed $v_{\max}$ of client in the geographical region.

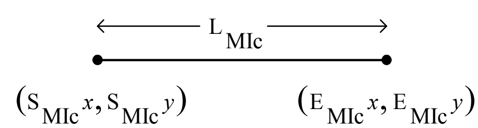


Fig. 3. Current Moving Interval

Let $v_c$ be the velocity in current moving interval $MI_c$, $L_{MIc}$ be the distanced traveled in $MI_c$ along direction $\theta_c$ and ($S_{MIc}x$, $S_{MIc}y$) and ($E_{MIc}x$, $E_{MIc}y$) be the starting and end point of $MI_c$ respectively (see Fig. 3). Assuming that the velocity $v_c$ remains same in the next MI also, we can get a circular predicted region with radius $L_{MIc}$ and centre ($E_{MIc}x$, $E_{MIc}y$).

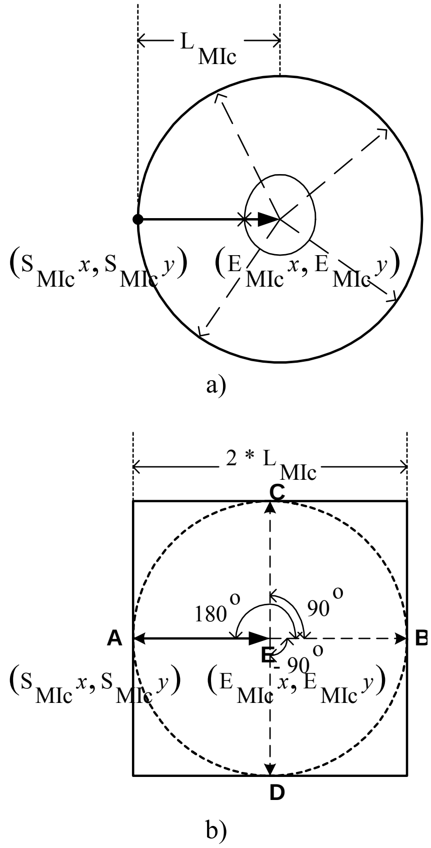


Fig. 4. Predicted Region a) Circular b) Square

Fig. 4 (a) shows the pictorial view of it. Since it is highly complex to take all the points on the circle in order to trap the region, we consider the following cases:

- user travels in same direction as $\theta_c$ in new MI.
- user travels in direction opposite to the $\theta_c$ in new MI, i.e. $\theta_c + 180°$ degrees, and
- user travels in direction perpendicular to $\theta_c$ in new MI, i.e. $\theta_c \pm 90°$ degrees.

Fig. 4 (b) shows the four point on the circle with $\theta_c = 0°$ and shows the predicted region that can be formed using the four points, i.e. square with side $2*L_{MIc}$. The advantage of this square region is that it is simple to generate it and has predicted region greater than that of circular one. So a user can reach points B, A, C and D in the above worst case at the end of next MI with velocity $v_c$, and direction $\theta_c$, $\theta_c + 180°$, $\theta_c + 90°$and $\theta_c$ -90° respectively. We define a set of vector sets which bound the predicted region as *Pred_Reg*={$A_v$,$B_v$,$C_v$,$D_v$,$E_v$}. Each vector set is associated with it's originating x-y coordinate of each of its vector, angular range within which those vectors are valid and maximum length the vectors should have to ensure that they lies in the predicted region. Therefore,vector sets $A_v$, $B_v$, $C_v$, $D_v$, and $E_v$ w.r.t points A,B,C,D,and E respectively,can be represented as follows:

$A_v = \{(A_x,A_y),(\theta_c-90^o, \theta_c + 90^o), L_{MIc}\}$
$B_v = \{(B_x,B_y),(\theta_c + 90^o, \theta_c-90^o), L_{MIc}\}$
$C_v = \{(C_x,C_y),(\theta_c + 180^o, \theta_c), L_{MIc}\}$
$D_v = \{(D_x,D_y),(\theta_c, \theta_c + 180^o), L_{MIc}\}$, and
$E_v = \{(E_x,E_y),(0^o, 360^o), L_{MIc}\}$

We will consider an example as shown in Fig. 5, to show use of vector sets in *Pred_Reg*.

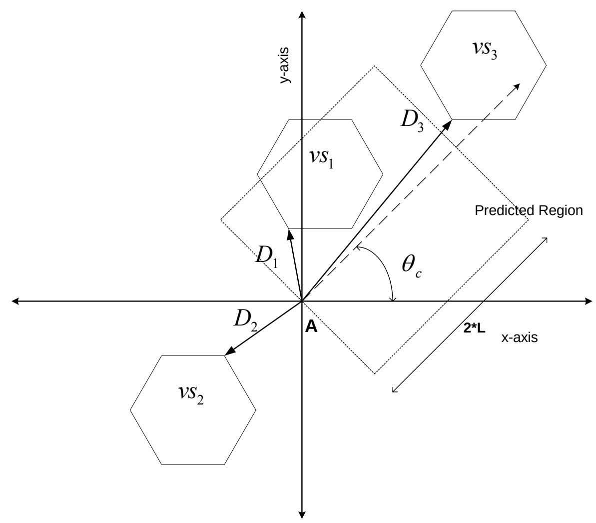


Fig. 5. Use of vector set in Pred_Reg

Considering the vector set $A_v$ for point ($A_x$,$A_y$) given as: $A_v = \{(A_x,A_y),(\theta_c-90^o, \theta_c + 90^o), L\}$
Calculating the distance vector $D_1$,$D_2$,$D_3$ of valid scopes $vs_1$,$vs_2$ and $vs_3$ respectively with respect to $A_v$ , it is clear from figure 5 that $D_1$ *and* $D_3$ satisfy the angular range of $A_v$ . But out of $D_1$ *and* $D_3$ only length of $D_1$ is within L. Hence valid scope $vs_1$ is considered as the valid scope within the predicted region by $A_v$ . Similar approach is followed for other vectors in *Pred_Reg* .

Now, we define our cost based cache replacement policy PRRP which takes into consideration access probability, data distance taking into consideration the predicted square region, valid scope area and data size in cache. Associated with each cached data object is the replacement cost. When a new data object needs to be cached and there is insufficient cache space, the object(s) with lowest replacement cost is removed until there is enough space to cache new object. For each valid scope in client's cache, its distance is calculated with respect to the vectors in *Pred_Reg* The cost of replacing a data value j of data item i in client's cache is calculated as:

$$Cost_{i,j} = \begin{cases} \dfrac{P_i . A(vs'_{i,j})}{S_{i,j}} . \underset{\forall p \in Pred\_Reg}{maximum} \left\{ \dfrac{1}{D(vs'_{i,j}, p)} \right\} & if \quad vs'_{i,j} \in Pred\_Reg \\ \dfrac{P_i . A(vs'_{i,j})}{S_{i,j}} . \dfrac{1}{D(vs'_{i,j})} & if \quad vs'_{i,j} \notin Pred\_Reg \end{cases}$$

where $P_i$ is the access probability of data item $i$, $A(vs'_{i,j})$ is the area of the attached valid scope $vs'_{i,j}$ for data value $j$, $S_{i,j}$ is the total size of data value $j$ and valid scope $vs'_{i,j}$, $D(vs'_{i,j}, p)$ is the distance between $vs'_{i,j}$ and point $(p_x , p_y)$ associated with vector set $p$ and $D(vs'_{i,j})$ is the distance between the current location of client and the valid scope $vs'_{i,j}$ (i.e. data distance) . The advantage of using this predicted region is that it dynamically changes with speed of client and Moving Interval and also takes into account the random movement of client.

Access probability for each data item is estimated by using exponential ageing method [5,6]. Two parameters are maintained for each data item $i$: a running probability $P_i$ and the time of the last access to item $t^l_i$. $P_i$ is initialized to 0. When a new query is issued for data item $i$, $P_i$ is updated using the following formula:

$$P_i = \alpha / (t_c - t^l_i) + (1-\alpha) P_i$$

where $t_c$ is the current system time and α is a constant factor to weigh the importance of most recent access in the probability estimate.

For calculating data distance between valid scope (either a polygon or a circle) and current location/point we select a reference point for each valid scope and take the distance (Euclidean distance) between the current location/point and the reference point. For polygonal valid scope, the reference point is defined as the endpoint that is closest to the current location/point and for circular valid scope, it is defined as the point where the circumference and the line connecting the current location and the center of the circle meet.

## V. SIMULATION MODEL

This section describes the simulation model used to evaluate the performance of the proposed PRRP. We compare PRRP with existing cache replacement algorithms such as PAID, LRU, FAR , Euclidean and Manhattan [14]. Our Simulator is implemented in C++.

### A. System Execution Model

Since seamless hand-off from one cell to another is assumed, the network can be considered a single, large service area within which the clients can move freely and obtain location dependent information services. In our simulation, the service area is represented by a rectangle with a fixed size of *Size*. We assume a "wrapped-around" model for the service area. In other words, when a client leaves one border of the service area, it enters the service area from the opposite border at the same velocity. The database contains *ItemNum* items. Every item may display *ScopeNum* different values for different client locations within the service area. Each data value has a size of *DataSize*. In the simulation, the scope distributions of the data items are generated based on voronoi diagrams (VDs) [12]. In our simulation, a scope distribution ( Fig. 6) contains 110 points randomly distributed in a square Euclidean space. The model assumes that two floating-point numbers are used to represent a two-dimensional coordinate and one floating-point number to represent the radius. The size of a floating-point number is *FloatSize*. The wireless network is modeled by an uplink channel and a downlink channel. The uplink channel is used by clients to submit queries, and the downlink channel is used by the server to return query responses to target clients. The communication between the server and a client makes use of a point-to-point connection. It is assumed that the available bandwidth is *UplinkBand* for the uplink channel and *DownlinkBand* for the downlink channel.

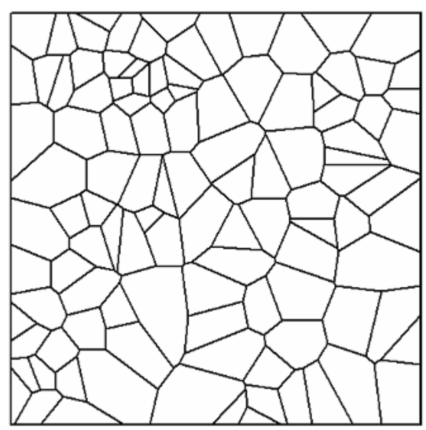

Fig. 6. Scope Distribution for performance evaluation: (ScopeNum=110)

### B. Client Execution Model

The mobile client is modeled with two independent processes: query process and move process. The query process continuously generates location-dependent queries for different data items. After the current query is completed, the client waits for an exponentially distributed time period with a mean of *QueryInterval* before the next query is issued. The client access pattern over different items follows a Zipf distribution with skewness parameter $\theta$. To answer a query, the client first checks its local cache. If the data value for the requested item with respect to the current location is available, the query is satisfied locally. Otherwise, the client submits the query and its current location uplink to the server and retrieves the data through the downlink channel.The move process controls the movement pattern of the client using the parameter *MovingInterval*. After the client keeps moving at a constant velocity for a time period of *MovingInterval*, it changes the velocity in a random way: the next moving direction (represented by the angle relative to the x axis, counter clock wise taken as positive) is selected randomly between 0 and 360, and the next speed is selected randomly between *MinSpeed* and *MaxSpeed*.

When the value of *MovingInterval* is small, the client's movement is rather random; when the value of *MovingInterva*l is large, the movement of the client behaves more like a pre-defined trip which consists of long straight-line segments. The client is assumed to have a cache of fixed size, which is a *CacheSizeRatio* ratio of the database size.

### C. Server Execution Model

The server is modeled by a single process that services the requests from clients. The requests are buffered at the server if necessary, and an infinite queue buffer is assumed. The FCFS service principle is assumed in the model. To answer a location-dependent query, the server locate the correct data value with respect to the specified location. Since the main concern of this paper is the cost of the wireless link, which is more expensive than the wired-link and disk I/O costs, the overheads of request processing and service scheduling at the server are assumed to be negligible in the model.

## VI. Performance Evaluation

In this section, the proposed PRRP cache replacement policy is evaluated against PAID, LRU, FAR, Euclidean and Manhattan using the simulation model described in the previous section. Table 1 shows the default parameter settings of the simulation model.

For our evaluation purpose, we assume that all data items follow the same scope distribution in a single set of experiments. Though, items can have different sets of distributions, but it is not possible to show all *ScopeNum* distributions. The results are obtained when the system has reached the stable state, i.e., the client has issued at least 20,000 queries, so that the warm-up effect of the client cache is eliminated. The CEB cache invalidation policy is employed for cache management.

Fig. 7, compares the improved performance of PRRP with existing cache replacement algorithms with respect to change in mean Query Interval. We observe that the performance of PRRP is better than all. As the query interval increases, cache hit ratio decreases, because the client would make more movements between two successive queries, thus has low probability to remain in the same valid scope queried previously when a new query is issued. PRRP obtains better performance than PAID with an average improvement of 5 percent.

TABLE I
CONFIGURATION PARAMETERS AND DEFAULT PARAMETER SETTINGS FOR SIMULATION MODEL

| Parameter | Description | Setting [a] |
|---|---|---|
| *Size* | size of the rectangle service area | 4000m*4000m |
| *ItemNum* | number of data items in the database | 500 |
| *ScopeNum* | number of different values at various locations for each item | 110 |
| *DataSize* | size of a data value | 128 bytes |
| *UplinkBand* | bandwidth of the uplink channel | 19.2 kbps |
| *DownlinkBand* | bandwidth of the downlink channel | 144 kbps |
| *F loatSize* | size of a floating-point number | 4 bytes |
| *QueryInterval* | average time interval between two consecutive queries | 50.0 s |
| *MovingInterval* | time duration that the client keeps moving at a constant velocity | 100.0s |
| *MinSpeed* | minimum moving speed of the client | 1m $s^{-1}$ |
| *MaxSpeed* | maximum moving speed of the client | 2m $s^{-1}$ |
| *CacheSizeRatio* | ratio of the cache size to the database size | 10 % |
| $\theta$ | skewness parameter for the Zipf access distribution | 0.5 |

Fig. 8, shows the effect of Moving Interval (varied from 50 seconds to 400 seconds) on replacement policies. The longer the moving interval, the less frequently the client changes velocity and direction and, hence, less random is the client’s movement. For small MI, the randomness in client movement is more as compared to larger MI. PRRP performs better than all policies for both small and large MI. The predicted region in PRRP helps to keep the data items within the influence of client’s movement, thereby reducing the affect of randomness of client. Average improvement of PRRP is 5 percent over the next best policy PAID. As the MI is increased the performance decreases. Because for relatively longer MI, a larger average distance difference is observed for two successive queries, which implies that client has a higher possibility of leaving certain regions. Consequently, the cached data are less likely to be reused for subsequent queries, which lead to a worse performance.

Effect of cache size on performance of replacement policies are shown in Fig. 9.As desired, the performance of replacement policies improves with increase in cache size. PRRP consistently out perform the existing policies.

Client cache hit ratio is shown against client speed in Fig. 10. Four speed ranges [15], 1~5m/s, 6~10m/s, 16~20m/s, 25~35m/s, corresponding to the speed of a walking human, a running human, a vehicle with moderate speed and a vehicle with high speed, respectively are used. It can be seen that very high cache hit ratio can be achieved for walking human. For higher speed range, the cache hit ratio drops as because client spend less time at each geographic location and the valid scope of each data item stored in cache becomes less effective. In PRRP, higher the speed of client, greater is the predicted region and hence more data items stored in the cache are held in that region. The over all performance of PRRP is best with respect to other policies in each speed range.

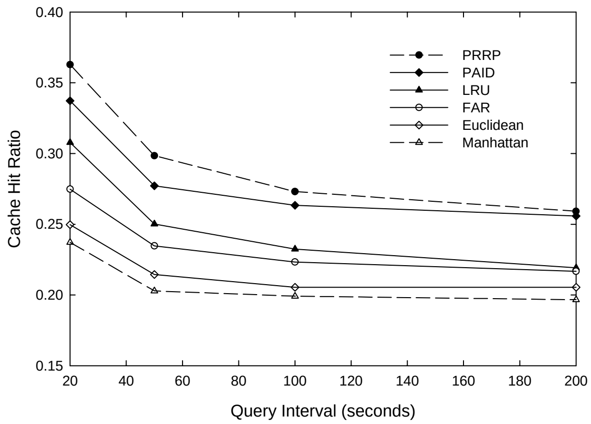


Fig. 7. Cache Hit Ratio vs Query Interval

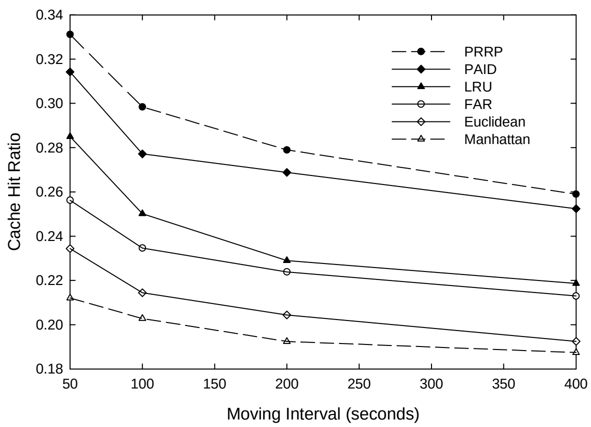


Fig. 8. Cache Hit Ratio vs Moving Interval

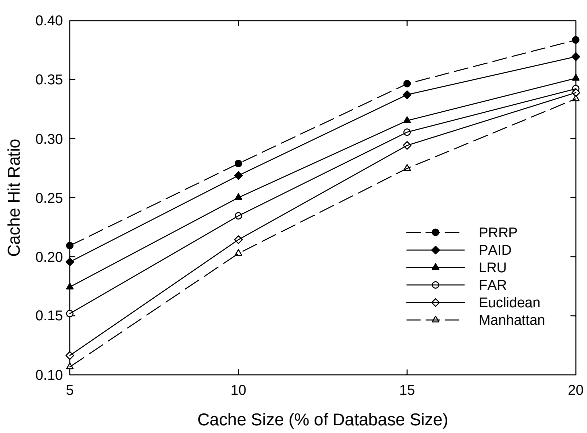


Fig. 9. Cache Hit Ratio vs Cache Size

The Zipf parameter $\theta$ determines the "skewness" of the access pattern over data items. When $\theta$=0, the access pattern is uniformly distributed. When $\theta$ increases, more access is focused on few items (skewed). Fig. 11 shows the impact of access pattern on performance of replacement policies. As desired, performance of PRRP along with other replacement policies increases with increase in $\theta$.

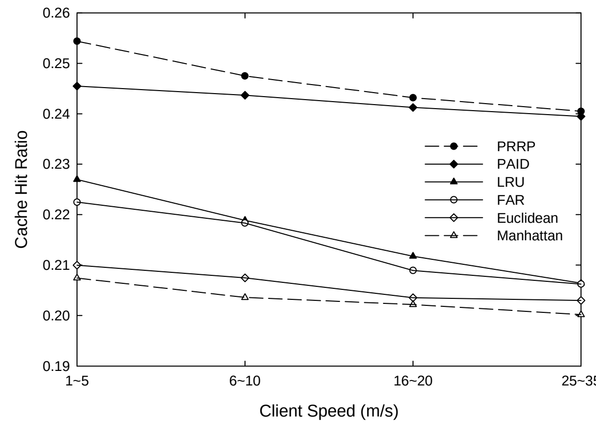


Fig. 10. Cache Hit Ratio vs Client Speed

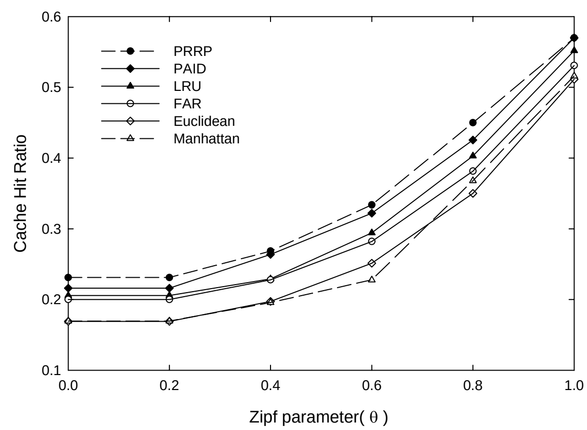


Fig. 11. Cache Hit Ratio vs Zipf parameter (θ)

## VII. Conclusion

In this paper, we proposed a cost based cache replacement policy PRRP which takes into consideration access probability, data distance taking into consideration the predicted square region, valid scope and data size in cache. Predicted region plays an important role in improving the system performance. Simulation results show that PRRP achieves on average of 5 percent improvement over next best PAID policy.

As for future work we are planning to study prefetching and data dissemination schemes for LDIS.

## Acknowledgment

The authors thank Baihua Zheng for her kind responses to queries related to this work and Intel PlanetLab, I.I.T. Roorkee, India, for providing support for this work.

## REFERENCES


[1] D. Barbara, “Mobile Computing and Databases: A Survey”, *IEEE Trans. on Knowledge and Data Engg*, 11(1), 1999.

[2] D. Barbara and T. Imielinski, “Sleepers and Workaholics: Caching Strategies in Mobile Environments”, *In Proc. of SIGMOD,* 1994.

[3] I.A. Getting, “The Global Positioning System”, *IEEE Spectrum*, 12(30), 1993.

[4] D.L. Lee, W.-C. Lee, J. Xu, and B. Zheng, “Data Management in Location-Dependent Information Services”, *IEEE Pervasive Computing*, 1(3), 2002.

[5] B. Zheng, J. Xu and D. L. Lee, “Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments”, *IEEE Trans. on Comp.*, 51(10), 2002.

[6] L. Yin, G. Cao and Y. Cai, “A generalized Target-Driven Cache Replacement Policy for Mobile Environments”, In Proceedings of the IEEE Symposium on Applications on the Internet, January 3003 ,pp. 14-21.

[7] K. Lai, Z. Tari and P. Bertok, “Location–Aware Cache Replacement for Mobile Environments”, *IEEE Globecom* 2004, pp. 3441-3447

[8] M. Erwig, R. H. Guting, M. Schneider and M. Vazirgiannis, “Spatio-Temporal Data Types: An Approach to modeling and Querying Moving Objects in Databases”, GeoInformatica,3(3), 269-296, 1999.

[9] Q. Ren and M.H. Dhunham, “Using Semantic Caching to Manage Location Dependent Data in Mobile Computing”, *ACM/IEEE MobiCom*, 210-221, 2000.

[10] S. Dar, M.J. Franklin, B.T. Jonsson, D. Srivastava, and M. Tan, “Semantic Data Caching and Replacement”, *VLDB*, 330-341,1996.

[11] A. Balamash and M. Krunz, “An Overview of Web Caching Replacement Algorithms”, IEEE Communications Surveys & Tutorials, 6(2), 2004.

[12] J. O’ Rourke, Computational Geometry in C, chapter 5, Univ. of Cambridge Press, 1998.

[13] E. O’Neil and P. O’Neil, “The LRU-k page replacement algorithm for database disk buffering”, Proceedings of the ACM SIGMOD, 296-306,1993

[14] Jung, I., You, Y., Lee, J., Kim, K.: Broadcasting and caching Policies for Location-Dependent Queries in Urban Areas. ACM WMC’02 (2002) 54-60

[15] Lu, C., Xing, G., Chipara, O., Fok, C.L.,: MobiQuery: A SpatioTemporal Data Service for Sensor Networks. Proceedings of the 2nd International Conference on Embedded Networked Sensor System, USA (2004) 320-334