

A Distance-based Predicted Region Policy for Cache Replacement in Mobile Environments

Kamal A. EIDahshan
Department of Mathematics
Faculty of science
Al-Azhar University-Cairo, Egypt

Ahmed A.A. Gad-ElRab
Department of Mathematics
Faculty of Science
Al-Azhar University-Cairo, Egypt

Ahmed Sobhi
Department of Mathematics
Faculty of Science
Al-Azhar University-Cairo, Egypt

ABSTRACT

Mobile computing has developed during recent years. Location dependent services are most popular services that the mobile environments support. Data caching is a critical issue that plays an important role in improving these services and system performance. In mobile environments, due to the limited cache size of mobile devices, the main problem in data caching is cache replacement which is finding a suitable subset of items for eviction from cache. In this paper, to solve this problem, A Distance-Based Predicted Region Policy for Cache Replacement in Mobile Environments is proposed. The proposed policy uses the root-mean squared distance that based on the distance between a client current location and the locations of each object whose data is cached for predicted region. Simulation results show that the proposed policy improves the system performance compared to existing schemes.

Keywords

Mobile computing, Data dissemination, location-dependent data, cache replacement, predicted region, DPRP.

1. INTRODUCTION

The fast development of wireless communication systems and advancement in computer hardware technology has led to the research area called mobile computing. The mobile computing research area includes the effect of mobility on hardware, software, users, data and computing in computer applications. It is an umbrella term used to describe technologies that enable people to access network services from anywhere and at any time. Without doubt, mobile computing is becoming an important part of people daily life [1], [2].

However, mobile environments properties as limited bandwidth, limited client power, frequent network disconnections, asymmetric communication, and limited client capacities led to many challenging problems for mobile data applications [3], [4].

In mobile environments, a mobile client can move and access information ubiquitously (anywhere and at any time). So, to enable mobile data access there are several methods. One of the main uses of wireless data is data dissemination. Data dissemination refers to broadcasting of database items to mobile clients through one or more wireless channels.

There are three mechanisms of data dissemination: *push-based Mechanism*, *On-demand (or pull-based) Mechanism*, and *Hybrid Mechanism*. In the *push based mechanism*, a server disseminates information generally without any request from clients. For example, advertising, weather reports, and news reports scenarios. In the *On-demand mechanism*, a server disseminates information based on the outstanding requests submitted by clients. For example, finding the nearest restaurant, buying a music album, or bank account activity

scenarios. In the *hybrid mechanism*, push based and on demand data are combined to complement each other. For example, advertising and selling music albums scenario. The advertisements are pushed and the mobile devices pull for buying the album.

The important two optimization issues of data dissemination are: *Minimizing access time* (i.e. access latency) and *Minimizing tuning time*. Access time is the period of time elapsed from the moment a mobile client requesting a data item(s) to the moment when the requested data item(s) is received by the client. While the tuning time is the time that a client spends actively listening to the broadcast wireless channel to receive the requested data items [5], [6], [7].

Client data caching is a common technique for minimizing access time, it is an effective technique to reduce access time by caching of frequently accessed data item on client side. When the client issues a query, it first searches the cache. If there is a valid copy of requested data in the cache, an answer is returned immediately. Otherwise, the client attempts to obtain the data item from a server.

In mobile environments, client data caching is much more desirable due to the limitations of mobile environments. Therefore, there are two common issues involved in client cache management: *Cache Invalidation* and *Cache Replacement*. A cache invalidation policy maintains data consistency between client's cache and the server database. While cache replacement policy determines which suitable subset of data items is replaced when it does not have enough free space to store a new data item [1], [2], [6], [8].

Due to the limitations of cache size on mobile devices, there is a need to design an efficient cache replacement algorithm to find a suitable subset of data items for eviction from the cache. Also, the design of an efficient cache replacement policy becomes very crucial and challenging to ensure good cache performance [9], [10].

Several cache replacement policies have been proposed in the literature. Most of these cache replacement policies are not suitable if the client changes its movement's direction quite often. To find most suitable cache replacement scheme, it is very important to take into account a current location of a client and the locations of each cached data for getting a predicted region of a client and a cached data. In this paper, A Distance-Based Predicted Region Policy for Data Cache Replacement in Mobile Environments (DPRP) is proposed. The proposed scheme uses the root-mean squared distance that is based on the distance between the client's current location and the locations of each cached data for a predicted region [2], [10], [11].

The rest of this paper is organized as follows: section 2 describes the related work. Section 3 describes the mobile system model. Section 4 describes the enhanced predicted region. In Section 5; the simulation model, the experimental results and analysis are presented. Conclusion and further work is given in Section 6.

2. RELATED WORK

Several location dependent cache replacement policies have been proposed. Most of these policies use cost functions, which take into account both the spatial and temporal properties of the client's movement, that incorporates different factors considered in cache replacement policies including access probability, valid scope area, data distance and data size. Access probability, is considered to be the most important factor that affects cache performance. Data with the least access probability will have the highest priority to be replaced by the new data object. Valid scope area refers to the geometric area of the valid scope of a data value. The larger the valid scope area of the data, the higher the probability that the mobile client requests this data. This is because, generally, the mobile client has a higher chance of being in large regions than small regions. Data distance refers to the distance between the current location of a mobile client and the valid scope of a data value. In a location-dependent data service, the server responds to a query with the suitable value of the data item according to the client's current location. As such, when the valid scope of a data value is far away from the client's current location, this data will have a lower chance to become usable again since it will take some time before the client enters the valid scope area again. Data size refers to the size of the data stored in the mobile client's cache. The amount of space required to store the data item in the cache is used to select an item for replacement, so keeping smaller size data items in the cache helps to accommodate a large number of data items [1], [2], [10], [11].

The Manhattan Distance-based cache replacement policy considers the distance between a client's current location and the location of each cached data item when there is need of cache replacement. The data items with the highest Manhattan distance are replaced. The Manhattan policy is limited because it considers the distance and spatial properties only. While the temporal properties and the direction of the client movement is not taken into account when making cache replacement decisions [11], [12].

The Farther Away Replacement (FAR) replacement policy considers the current location and direction of the mobile client to make the replacement decision. The replacement strategy is based on the fact that the data which are not in the moving direction and farthest away from the user will not be visited in the near future. Based on the direction of movement, the data is arranged as two sets, In-Direction and Out-Direction. To replace data the Out-Direction set is considered first, when it is empty the furthest segment in the In-Direction set will be replaced. FAR considers only the spatial properties for cache replacement and the temporal properties are not taken into account, it is also not very useful when the mobile client frequently changes its direction (random movement) [11], [13].

Probability Area (PA) policy considers only temporal property of data for replacement, the cost function for replacement is formed by considering the parameters access probability and valid scope area. The data with low access probability and a small valid scope area is evicted first. PA does not take into account the data distance and the size of the

data object stored in cache. PA is ineffective when client changes its direction frequently (random movement) [1], [11].

Probability Area Inverse Distance (PAID) policy considers both spatial and temporal properties of data for replacement. The cost function for replacement is formed by considering the parameters access probability, valid scope area and data distance. The data with low access probability, a small valid scope area, and a long distance is evicted first. PAID considers temporal and spatial property, does not take into account the size of the data object stored in cache, considers only the clients current movement direction, and ineffective when client changes its direction frequently (random movement) [1], [11].

Mobility Aware Replacement Scheme (MARS) is also a cost based policy. It makes cache replacement decisions through a cost function which takes into account both the spatial and temporal properties of the client's movement. The cost function consists of the client location, movement's direction and access probability. The data item with the lowest value for the cost function is removed from the client's cache and replaced by the new object. MARS is not very useful when mobile client frequently changes its direction (random movement) [11], [14].

None of these cache replacement policies is suitable if the client changes its movement's direction quite often. Existing cache replacement policies only consider the data distance (directional/undirectional) but not the distance based on region or area where the client may be in the near future. In Prioritized Predicted Region based Replacement Policy (PPRRP) instead of taking the direction of client's movement they predict an area/region in which the client will be in the near future while selecting an item for replacement and the data items in the predicted region are not removed from the cache. PPRRP tried to get the benefit of both temporal and spatial properties, and it is also suitable when the mobile client frequently changes its movement's direction (random movement).

Associated with each cached data item is the replacement cost; the cost function in PPRRP is calculated based on the access probability, valid scope area, data size in cache and distance of data based on the predicted region. When a new data item needs to be cached and there is insufficient cache space, the data item out of the predicted region with lowest value for cost function is removed from the client's cache and replaced by the new data item [10], [11].

3. MOBILE SYSTEM MODEL

A mobile system consists of two distinct sets of entities: mobile clients and fixed hosts (see Figure 1). A mobile client (MC) is a mobile unit which is capable of connecting to fixed network via a wireless link (or wireless channel). Fixed networks are classified as either fixed hosts or base stations, and connected together via a fixed high-speed wired network. Base stations (BS) are gateways of the mobile units to the fixed network, they are equipped with a wireless interface and offer network access services of which mobile units are clients. A fixed host (data server) is a computer in the fixed network which is not capable of connecting to a mobile unit.

The wireless channel between MC and BS 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 a BS, while the downlink channel is used by BSs to disseminate information or to forward the answers from the server to a target client.

To manage the units' mobility, the entire geographical area is divided into one or more domains called cells, each of which is supported by at least one base station. An MC movement from one cell to another, while retaining its wireless connection is called hand-off. After hand-off, its wireless connection is switched to the new cell [1], [15].

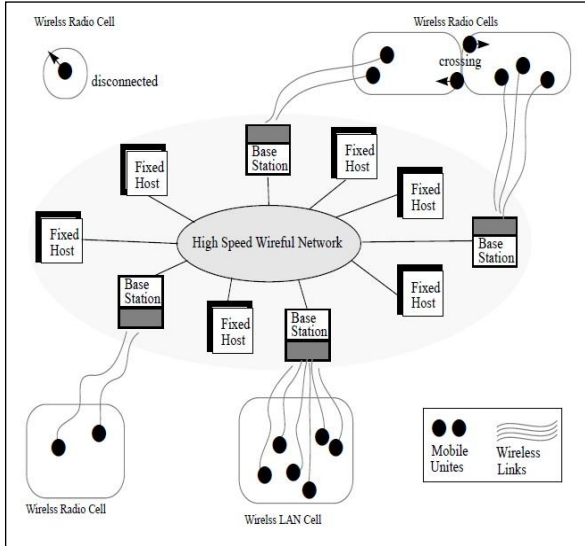


Fig 1: A Mobile System Model [15]

The mobile system model provides location dependent services to mobile clients. When the client issues a query, a data item can show different values when it is queried from different locations. Note that, data item and data value are different. A data item value for a data item is valid for a certain geographical area (valid scope). The valid scope of an item value is defined as the area (set of cells) within which the data item value is valid. For example (see Figure 2), the “nearby-restaurant” is a data item, the data item value for this data item is {A} with the valid scope {1, 2} or {B} with the valid scope {3, 4}. Note that, a data item value varies when it is queried from different locations [2], [10].

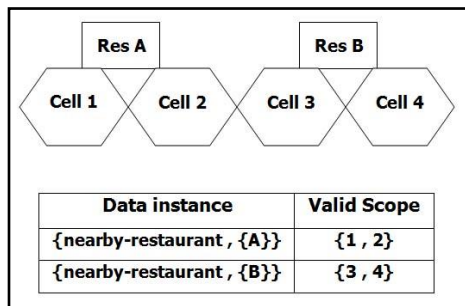


Fig 2: Data instance and valid scope representation.

4. AN ENHANCED PREDICTED REGION BASED CACHE REPLACEMENT

In this section, a new cache replacement scheme based on a predicted region is proposed. Firstly, a predicted region which is defined in PPRR [10] will be described. Secondly, the proposed distance-based predicted region will be introduced. Thirdly, the replacement cost function will be introduced. Finally, the proposed cache replacement scheme will be described.

4.1 Predicted region

Predicted regions play an important role in improving the system performance. One of the main advantages of predicted

regions is that unrestricted mobility is considered and the performance of the system that uses the predicted region is enhanced. Using the predicted region of user influence, the data values in the vicinity of the client's current position are not purged from cache.

The predicted region is based on current client's movement pattern. For random movement the duration between the change in direction and the velocity is known as the Moving Interval (MI). Let v_c be the velocity in current moving interval MI_c , L_{MIc} be the length of MI_c along direction θ_c (see Figure 3), and (x_s, y_s) and (x_e, y_e) be the starting and end points of MI_c respectively (see Figure 4). The predicted region of user presence in the near future is the circle with radius L_{MIc} and centre (x_e, y_e) (see Figure 5) [10], [11]. This predicted region is called prioritized predicted region (PPR).

The main disadvantage for this PPR is using length of MI_c as radius of predicted region because it will produce a huge computation overhead. MI requires to know a start point of MI, an end point of MI, the mobile node's velocity, and movement's direction.

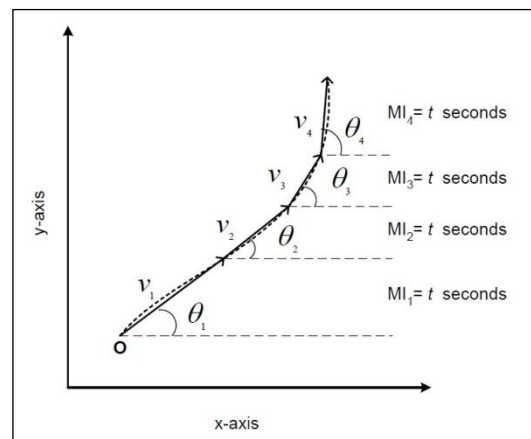


Fig 3: The discrete movement of a mobile user with MI of t seconds [10]

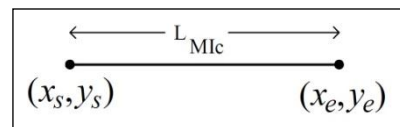


Fig 4: The starting and end point of MI_c [10]

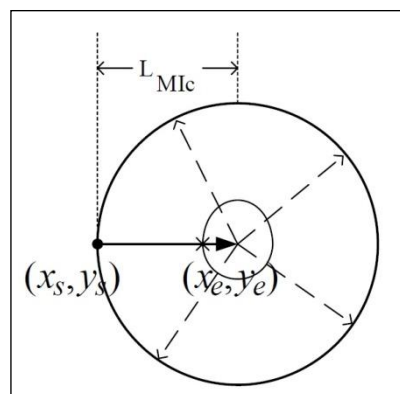


Fig 5: The predicted region [10]

4.2 Proposed distance-based predicted region

To avoid the disadvantages of PPR, a distance-based predicted region (DPR) will be proposed. With DPR, Instead of using

length of MI_C as radius of PPR, we will use the root-mean squared distance that is based on the distance between a client's current location and the locations of each object whose data is cached.

Let the radius r be calculated as the root-mean squared distance (see Figure 6). So, if the current location (center of the predicted region) is c , the radius of the predicted region is:

$$r = \sqrt{\frac{\sum_{i=1}^K (c_i - c)^2}{K}} = \sqrt{\frac{\sum_{i=1}^K (d_i)^2}{K}}$$

Where:

- r Radius of the predicted region.
- $(c_i - c)$ Distance (d_i) between the current location (c) and valid scope (c_i).
- K Number of object whose data is cached.

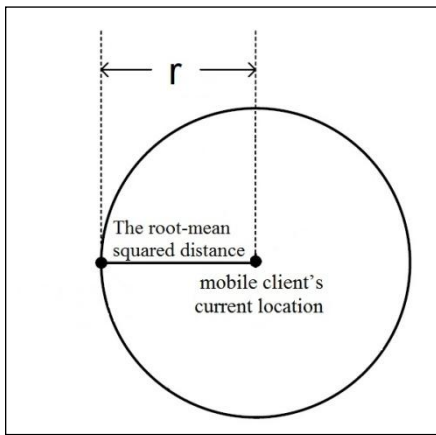


Fig 6: Distance-based predicted region

One of the advantages of using the DPR is that it reduces computation overhead and also takes into account the random movement of the client. Reducing the computation overhead is critical and plays an important role in improving the overall performance of cache replacement schemes. The cost function in DPR is calculated the same way as PPR. It is calculated based on the access probability, valid scope area, data size in the cache and the distance between the mobile client's current location and the locations of objects whose data is cached. The distance is calculated such that the data values within the predicted region are given higher priority than the data values outside the predicted region. When a new data item needs to be cached and there is insufficient cache space, the data value out of the predicted region with lowest value for cost function is removed from the client's cache and replaced by the new data value [10], [11].

4.3 The cost function for replacement

The cost of data value j of data item i in client's cache is calculated as [10], [11]:

$$Cost_{i,j} = \begin{cases} \frac{P_i \cdot A(vs'_{i,j})}{S_{i,j}} \cdot \frac{1}{\min\{r, D(vs'_{i,j})\}} & \text{if } vs'_{i,j} \in \text{DPR} \\ \frac{P_i \cdot A(vs'_{i,j})}{S_{i,j}} \cdot \frac{1}{D'(vs'_{i,j})} & \text{if } vs'_{i,j} \notin \text{DPR} \end{cases}$$

Where

- P_i The access probability of data item i .

- $A(vs'_{i,j})$ The area of the valid scope $vs'_{i,j}$ for data value j .
- $S_{i,j}$ The size of data value j and valid scope $vs'_{i,j}$.
- $D(vs'_{i,j})$ The distance of the valid scope $vs'_{i,j}$ from the current user position.
- $D'(vs'_{i,j})$ The distance of the valid scope $vs'_{i,j}$ from the centre of the predicted region.
- DPR The distance-based predicted region.

The distance of data items in the cache is calculated as follows:

- The distance of data items outside DPR is calculated from the centre of the circle.
- The distance of data items inside DPR is calculated as the minimum of $\{r, \text{distance of the valid scope from the current position of the user}\}$.

Calculating the distance of data items in this way ensures that:

- Items outside DPR always have the lower priority than those inside the predicted region.
- Items inside DPR are closer to the user and thus have higher priority.

4.4 The proposed cache replacement scheme

Here, based on DPR, a new scheme called a Distance-Based Predicted Region Policy for Data Cache Replacement in Mobile Environments (DPRP) is proposed. In DPRP, data is cached at the mobile client. When the client issues a query, it first searches its cache. If there is a valid copy in the cache, an answer is returned immediately. Otherwise, the client attempts to obtain the data value from the server and store it on its cache [16]. If the cache does not have enough free space to store a new data value, DPRP does the following procedure for the cached data items in the mobile client:

- Compute the access probability, valid scope area, data size and distance of data based on DPR.
 - Compute the radius (r) of DPR by using the root-mean squared distance.
 - Determine which data values inside or outside the DPR
 - If $d_i > r$ then the data value i outside DPR.
 - If $d_i \leq r$ then the data value i inside DPR.
- d_i is the distance between the client current location and the valid scope of data value i .
- Compute the cost function, $Cost_{i,j}$, for each data value j of each data item i that exists out DPR, the data item with lowest cost function value is removed from the client's cache and replaced by the new data item.

Algorithm 1 shows the steps of DPRP.

Algorithm 1: Steps of DPRP Algorithm

Mobile Client (MC) requests for data item D_i
if D_i is valid and in cache **then** validate and return D_i

```

else if cache misses  $D_i$  then
    send the request to the server
    get  $D_i$  from DB server
    server send  $D_i$  to MC
    if enough free space in cache then store  $D_i$  in cache
    else if not enough free space in cache then
        create enough space by replacing data value (s)
        from cache which:
            out of predicted region
            has lowest cost function value
    end if
end if

```

5. SIMULATION AND RESULTS

This section describes the simulation model which is used to evaluate the performance of DPRP. To show the effectiveness of DPRP, it is compared with PPRRP [10]. The discrete-time simulation package OMNeT++ [17] was used to implement this simulation model.

5.1 Simulation Model Description

The simulation model consists of three main entities: network entity, mobile client entity, and sever entity. These entities are described as follows [1].

(a) **Network entity** is a cellular network that consists of many cells and provides seamless handoffs from one cell to another when mobile clients move across different cells. As such, the network can be considered as a single large service area, the clients can move freely and obtain location-dependent information services. The service area is represented by a rectangle of fixed size. The database contains items and every item may display different values for different client locations within the service area. The wireless network is modeled by an uplink channel and a downlink channel. The uplink channel is used by the client to query the server and the downlink channel is used by the server to respond to the query.

(b) **Mobile client entity** is modeled with two independent processes: the query process and the move process. The query process continuously generates location-dependent queries for different data items. After the current query is completed, the client waits for a query interval, QI, before the next query is issued. To answer a query, the client's cache is checked first. If the data value for the requested item with respect to the current location is available, the query is satisfied locally. Otherwise, the client gives the query and its current location to the server and retrieves the data from the server through the downlink channel. The move process controls the client's movement pattern using the parameter MovingInterval (Table 1). After the client keeps moving at a constant velocity for a time period, it changes the velocity in a random way for next MI. The next speed is selected randomly between the parameter MinSpeed and the parameter MaxSpeed and the next moving direction (represented by the angle relative to the x axis) is selected randomly between 0 and 360. The client is assumed to have a fixed size cache, which is a ratio of the database size.

(c) **Server entity** is modeled by a single process that offers to the requests from clients. To answer a location-dependent query, the server locates the correct data value with respect to the specified location.

5.2 Performance Evaluation

In this subsection, PPRRP [10] and DPRP are evaluated by using the described simulation model. Table 1 shows the

default parameter settings of the simulation model. To compute the data distance between valid scope and current location, a reference point is selected for each valid scope and calculate the *Euclidean distance* between the current location and this reference point. The reference point is defined as the endpoint that is closest to the current location. To compute the access probability, two parameters are maintained for each data item i : a running probability (P_i) and the time of the last access to item (t_i'). Initially, P_i is set to 0. When a new query is issued for data item i , P_i is updated using the following formula:

Where, t_c is the current system time and α is a constant factor to weight the importance of most recent access in the probability estimation.

In the performance evaluation, "cache hit ratio", "access time" and "energy consumption" are used as performance evaluation metrics. The cache hit ratio can be defined as the ratio of the number of queries answered by the client's cache to the total number of queries generated by the client. The access time is the period of time elapsed from the moment a mobile client requests a data item to the moment when the requested data item is received by the client [1], [10].

Table 1. The default parameter settings of the simulation model

Parameter	Description	Setting
Size	Size of the rectangular service area	1000*1000 m
MCNum	Number of mobile clients in the service area	5
LNum	Number of location in the service area	25
ItemNum	Number of data items in the database	250
UplinkBand	Bandwidth of the uplink channel	1 Mbps
DownlinkBand	Bandwidth of the downlink channel	2 Mbps
Query Interval (QI)	average time interval between two consecutive queries	25 s
Moving Interval (MI)	Time duration that the client keeps moving at a constant velocity	50 s
MinSpeed	minimum moving speed of the client	1, 3 m s ⁻¹
MaxSpeed	maximum moving speed of the client	2, 6 m s ⁻¹
CacheSizeRatio	Ratio of the cache size of the database size	4% = 10
α	Weight factor for computing the access probability.	0.25

In the rest of this section, to prove the efficiency of the DPRP compared to the PPRRP [10], the effect of query interval, moving interval, cache size, number of queries, and number of clients by changing their values will be shown in details.

5.2.1 Effect of Changing the Query Interval

The query interval is the time interval between two consecutive client queries. In this set of experiments, the mean query interval was increased from 20 to 100 seconds. Table 2 shows the default parameters settings for changing the query interval.

Table 2. The default parameter settings for changing query interval.

Moving Interval (MI)	Cache Size	Number of clients
50 s	4%	5

Figure 7 shows the performance results of cache replacement policies for query interval versus the cache hit ratio. As shown in Figure 7, as the query interval increases, the cache hit ratio decreases. This is because, the client would make more movements between two successive queries, thus the client has a lower probability of residing in one of the valid scopes of the previously queried data items when a new query is issued. Consequently, the cached data are less likely to be re-used for subsequent queries. This leads to a decreased performance of the cache hit ratio with increase in the query interval. Also, the cache hit ratio of DPRP is higher than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

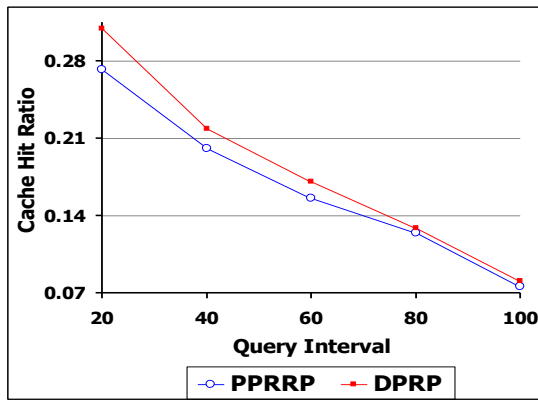


Fig 7: Cache hit ratio vs query interval

Figure 8 shows the performance results of cache replacement policies for query interval versus access time. As shown in Figure 8, as the query interval increases, the access time decreases. This is because, the client would make more movements between two successive queries, thus the client has a lower probability of residing in one of the valid scopes of the previously queried data items when a new query is issued. Consequently, the cached data are less likely to be re-used for subsequent queries. This leads to the access time increased as the query interval increased, but the total number of queries is decreased with MI and cache size not changed. For all these reasons, the access time decreased as the query interval increased. Also, access time of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

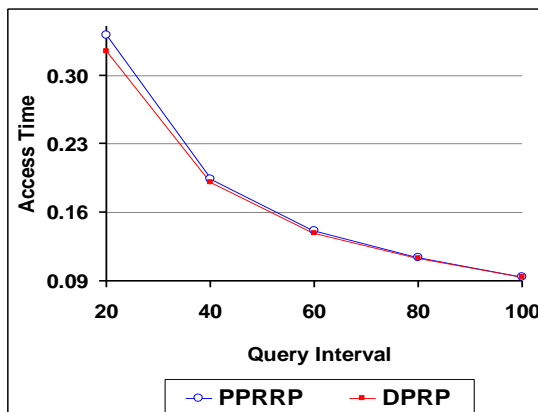


Fig 8: Access time vs query interval

Figure 9 shows the performance results of cache replacement policies for query interval versus energy consumption. As shown in Figure 9, as the query interval increases, the energy consumption decreases. This is because, the client would make more movements between two successive queries, thus the client has a lower probability of residing in one of the valid scopes of the previously queried data items when a new query is issued. Consequently, the cached data are less likely to be re-used for subsequent queries. This leads to the energy consumption increased as the query interval increased, but the total number of queries is decreased with MI and cache size invariant. For all these reasons, the energy consumption decreased as the query interval increased. Also, the energy consumption of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

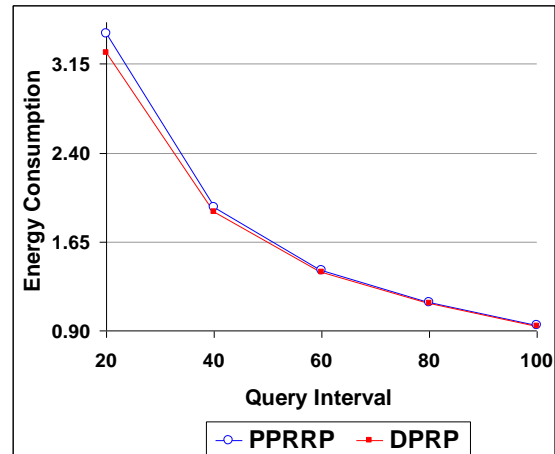


Fig 9: Energy consumption vs query interval

As shown, when the two cache replacement policies are compared, the DPRP has a better performance when the query interval is changed.

5.2.2 Effect of Changing the Moving Interval

This subsection examines the performance of the replacement policy when the moving interval is varied. MI is the time duration that the client keeps moving at a constant velocity and direction. In this set of experiments, the moving interval was varied from 25 seconds to 225 seconds. Table 3 shows the default parameters settings for changing MI.

Table 3. The default parameter settings for changing moving interval

Query Interval (QI)	Cache Size	Number of Queries	Number of clients
25 s	4%	200	5

Figure 10 shows the performance results of the cache replacement policies for MI versus cache hit ratio. As shown in Figure 10, as MI increases, the cache hit ratio decreases. This is because, there is a high probability of the client leaving one valid region and entering another. Consequently, the cached data are less likely to be re-used for subsequent queries. This leads to a decreased performance of cache hit ratio with increase in MI. Also, cache hit ratio of DPRP is higher than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

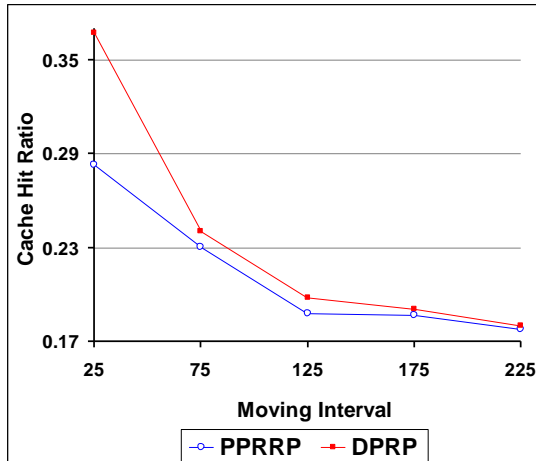


Fig 10: Cache hit ratio vs moving interval

Figure 11 shows the performance results of the cache replacement policies for MI versus access time. As shown in Figure 11, as MI increases, the access time increases. This is because, there is a high probability of the client leaving one valid region and entering another. Consequently, the cached data are less likely to be re-used for subsequent queries. This leads to, with query interval and cache size invariant, the access time increased as the MI increased. Also, access time of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

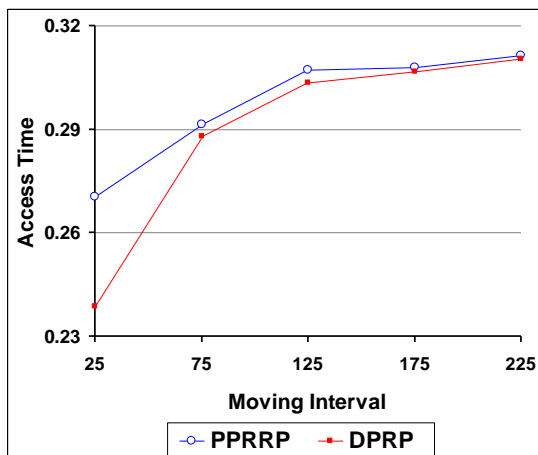


Fig 11: Access time vs moving interval

Figure 12 shows the performance results of cache replacement policies for MI versus energy consumption. As shown in Figure 12, as MI increases, the energy consumption increases. This is because, there is a high probability of the client leaving one valid region and entering another. Consequently, the cached data are less likely to be re-used for subsequent queries. This leads to, with query interval and cache size not changed, the energy consumption increased as the MI increased. Also, energy consumption of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

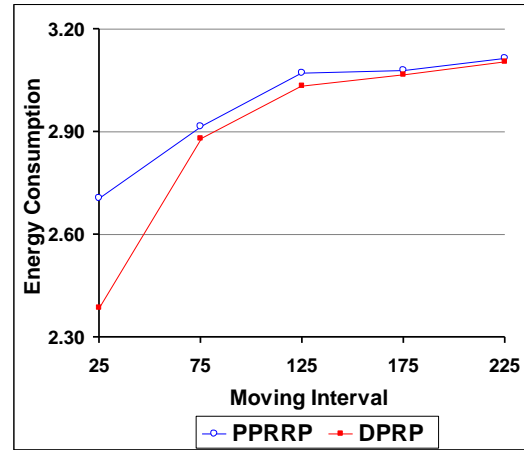


Fig 12: Energy consumption vs moving interval

For large MI, there is a less frequent change in the velocity of the client. Hence, there is less random client's movement. While, for small MI, the randomness in client movement is more as compared to larger MI. After the moving interval reaches 175 seconds, such an influence is eliminated and, hence, the performance becomes flat but when the two cache replacement policies are compared, the DPRP has a better performance for both small and large MI.

5.2.3 Effect of Changing the Cache Size

This subsection examines the performance of the replacement policy when the cache size is varied. In this set of experiments, the cache size was varied from 4% to 20%. Table 4 shows the default parameters settings for changing cache size.

Table 4. The default parameter settings for changing cache size

Query Interval (QI)	Moving Interval (MI)	Number of Queries	Number of clients
25 s	50 s	200	5

Figure 13 shows the performance results of cache replacement policies for the cache size versus the cache hit ratio. As shown in Figure 13, as the cache size increases, the cache hit ratio increases. This is because, the cache can hold a large number of data items. Consequently, the cached data are likely to be re-used for subsequent queries which increases the probability of getting a cache hit. This leads to an increased performance of cache hit ratio with the increase in cache size. Also, cache hit ratio of DPRP is higher than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

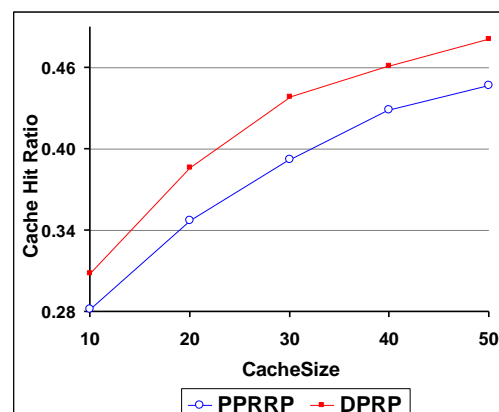


Fig 13: Cache hit ratio vs cache size

Figure 14 shows the performance results of cache replacement policies for the cache size versus the access time. As shown in Figure 14, as the cache size increases, the access time decreases. This is because, the cache can hold large number of data items. Consequently, the cached data are likely to be re-used for subsequent queries. This, with query interval and MI invariant, leads to the access time decreases as the cache size increases. Also, the access time of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

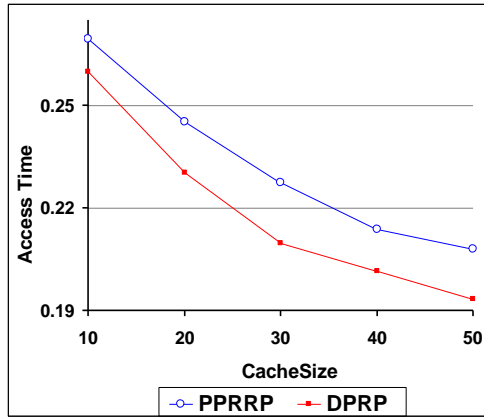


Fig 14: Access time vs cache size

Figure 15 shows the performance results of the cache replacement policies for the cache size versus the energy consumption. As shown in Figure 15, as the cache size increases, the energy consumption decreases. This is because, the cache can hold large number of data items. Consequently, the cached data are likely to be re-used for subsequent queries. This, with query interval and MI invariant, leads to the energy consumption decreases as the cache size increase s. Also, energy consumption of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

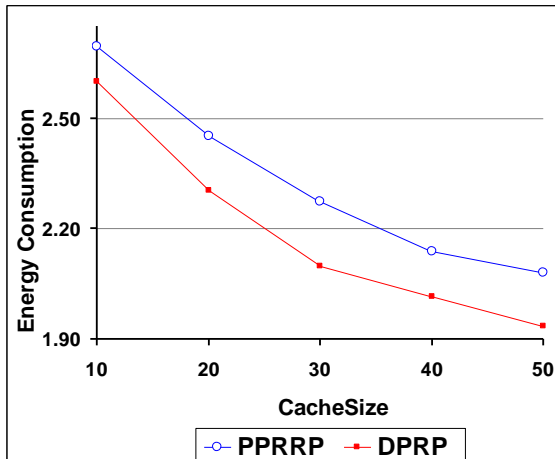


Fig 15: Energy consumption vs cache size

As shown, when the two cache replacement policies are compared, the DPRP has a better performance when the cache size is changed from small to large size cache.

5.2.4 Effect of Changing the Number of Queries

This subsection examines the performance of the replacement policy when the number of queries is varied. In this set of experiments, the number of queries was varied from 50 to 250. Table 5 shows the default parameters settings for changing the number of queries.

Table 5. The default parameter settings for changing number of queries

Moving Interval (MI)	Cache Size	Number of clients
50 s	4%	5

Figure 16 shows the performance results of cache replacement policies for the number of queries versus the cache hit ratio. As shown in Figure 16, as the number of queries increases, the cache hit ratio increases. This is because, with MI invariant, there is a high probability of the client would make more two successive queries one valid region, thus the cached data are likely to be re-used for subsequent queries which increase the probability of getting a cache hit. This leads to an increased performance of cache hit ratio with the increase in number of queries. Also, the cache hit ratio of DPRP is higher than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

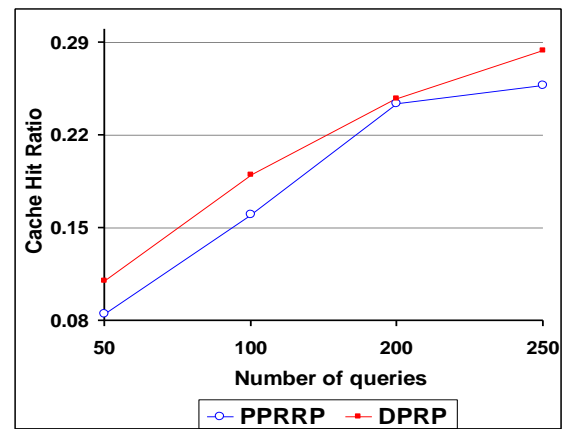


Fig 16: Cache hit ratio vs number of queries

Figure 17 shows the performance results of the cache replacement policies for a number of queries versus the access time. As shown in Figure 17, as the number of queries increases, the access time increases. This is because, with MI invariant, there is a high probability that the client would make more two successive queries on the valid region, thus the cached data are likely to be re-used for subsequent queries. This leads to the access time decreased as the number of queries increased, but the total number of queries is increased with MI when the cache size invariant. For all these reasons, the access time increased as the number of queries increased. Also, access time of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

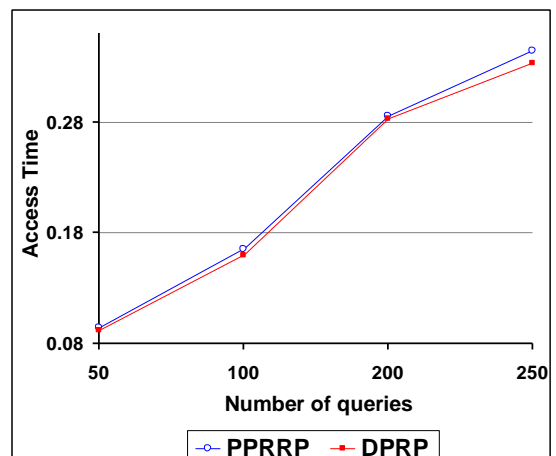


Fig 17: Access time vs number of queries

Figure 18 shows the performance results of the cache replacement policies for a number of queries versus the energy consumption. As shown in Figure 18, as the number of queries increases, the energy consumption increases too. This is because, with MI invariant, there is a high probability that the client would make more two successive queries on the valid region, thus the cached data are likely to be re-used for subsequent queries. This leads to the energy consumption decreases as the number of queries increases, but the total number of queries is increases with MI and cache size invariant. For all these reasons, the energy consumption increases as the number of queries increases. Also, the energy consumption of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

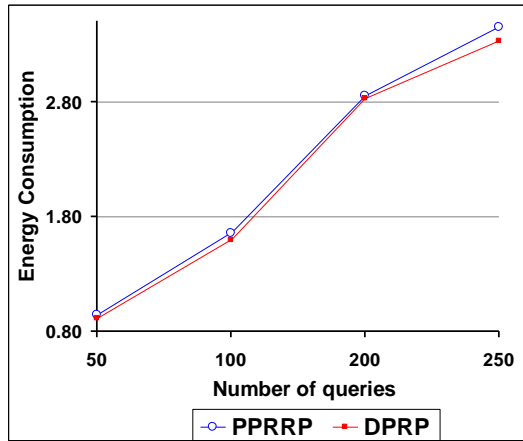


Fig 18: Energy consumption vs number of queries

As shown, when the two cache replacement policies are compared, the DPRP has a better performance when the number of queries is changed from a small to large number of queries.

5.2.5 Effect of Changing the Number of Clients

This subsection examines the performance of the replacement policy when the number of clients is varied. In this set of experiments, the number of clients was varied from 5 to 25 clients. Table 6 shows the default parameters settings for changing the number of clients.

Table 6. the default parameter settings for changing number of clients.

Query Interval (QI)	Moving Interval (MI)	Cache Size	Number of Queries
25 s	50 s	4%	80

Figure 19 shows the performance results of the cache replacement policies for the number of clients versus the cache hit ratio. As shown in Figure 19, as the number of clients increases, the cache hit ratio increases. This is because, the total number of queries is increased, and the total size of cache is increased. Consequently, the cache hit ratio is likely to be increased as the number of clients increased. Also, the cache hit ratio of DPRP is higher than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

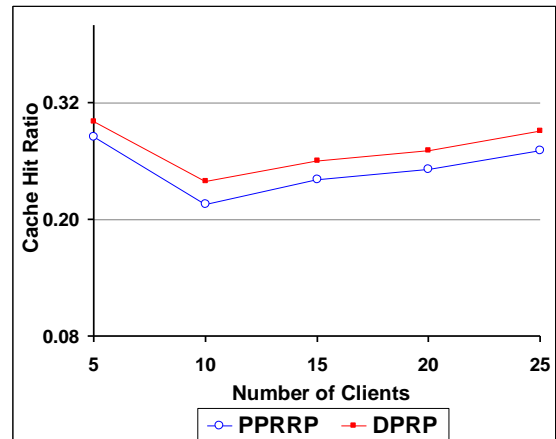


Fig 19: Cache hit ratio vs number of clients

Figure 20 shows the performance results of the cache replacement policies for a number of clients versus the access time. As shown in Figure 20, as the number of clients increases, the access time decreases. This is because, the total number of queries is increased, and the total size of cache is increased. Results show that, the increase in the number of clients and the cache size affect the access time more than the increase in the total number of queries. Consequently, the access time is likely to be decreased as the number of clients increases. Also, the access time of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

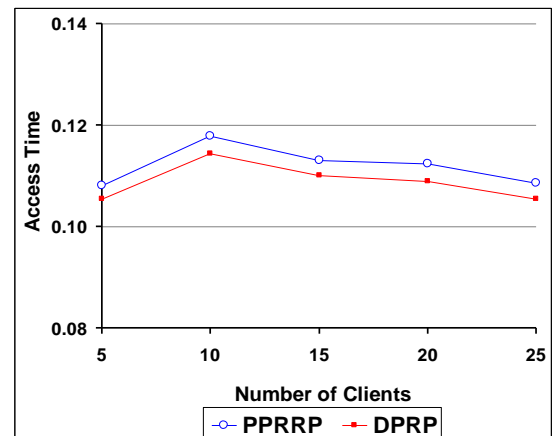


Fig 20: Access time vs number of clients

Figure 21 shows the performance results of the cache replacement policies for a number of queries versus the energy consumption. As shown in Figure 21, as the number of clients increases, the energy consumption decreases. This is because, the total number of queries increases, and the total size of cache increases. Results show that, the increases in the number of the clients and cache size are affect the energy consumption more than increases in the total number of queries. Consequently, the energy consumption is likely to decrease as the number of clients increases. Also, the energy consumption of DPRP is lower than its value in PPRRP. This is because DPRP uses DPR instead of PPR.

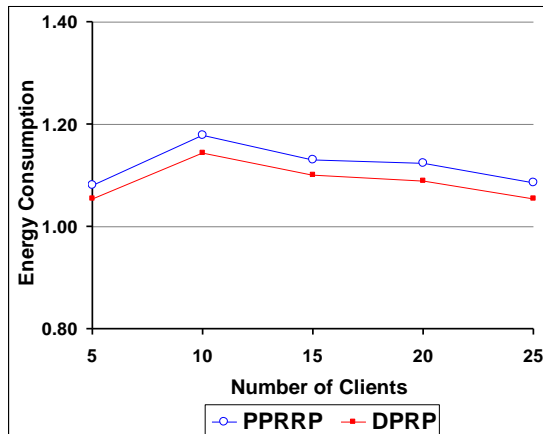


Fig 21: Energy consumption vs number of clients

Also, before the number of clients reaches 10 clients the cache hit ratio decreases and the total access time and the total energy consumption are increased. This is because, before the number of clients reaches 10 clients, the randomness in the client movement increases. After the number of clients reaches 10 clients, such an influence is eliminated and, hence, the performance of the cache hit ratio increases, the total access time and the total energy consumption decreases.

As shown, when the two cache replacement policies are compared, the DPRP has a better performance when the number of clients is changed from a small to a large number of clients.

6. CONCLUSION AND FURTHER WORK

In this paper, a distance-based predicted region policy for cache replacement is introduced (DPRP). DPRP is based on an enhanced predicted region and a cost function for selecting data items to be replaced from the cache. PPRRP uses MI as a radius of the predicted region. On the contrary, DPRP uses the root-mean squared distance that based on the distance between a client's current location and the locations of each cached data. In addition, DPRP uses the cost function that considers access probability, data distance, valid scope area and the data size in the cache in replacement. A number of simulation experiments have been conducted to evaluate the performance of the DPRP. The simulation results demonstrate that DPRP, with different system settings, gives better performance in comparison with PPRRP. In the further work, the use effect of the future location prediction accompanied with the cost function and the enhanced predicted region in improving the cache replacement issue will be studied.

7. REFERENCES

- [1] Zheng, B., Xu, J., and Lee, D. L. 2002. Cache invalidation and replacement strategies for location-dependent data in mobile environments. *Computers, IEEE Transactions on*, 51(10), 1141-1153.
- [2] Chavan, H., Sane, S., and Kekre, H. B. 2011. A Markov Model Based Cache Replacement Policy for Mobile Environment. In *Technology Systems and Management* (pp. 18-26). Springer Berlin Heidelberg.
- [3] Xu, J., Zheng, B., Zhu, M., and Lee, D. L. 2002. Research challenges in information access and dissemination in a mobile environment. In *Proceedings of the Pan-Yellow-Sea International Workshop on Information Technologies for Network Era*, 1-8.

- [4] Tabassum, K., Syed, M. Q., and Damodaram, A. 2011. Enhanced-Location-Dependent Caching and Replacement Strategies in Mobile Environment. *IJCSI Issues*, 8(4).
- [5] Drakatos, S., Pissinou, N., Makki, K., and Douligeris, C. 2006. A future location-prediction replacement strategy for mobile computing environments. In *Wireless Communications and Networking Conference, 2006. WCNC 2006. IEEE* (Vol. 4, pp. 2252-2260). IEEE.
- [6] Tabassum, K., Hijab, M., and Damodaram, A. 2010. A data mining approach for Cache Replacement in Location-dependent Environment. In *Computer Research and Development, 2010 Second International Conference on* (pp. 126-130). IEEE.
- [7] Katsaros, D., Nanopoulos, A., and Manolopoulos, Y. (Eds.). 2005. *Wireless Information Highways*. IGI Global.
- [8] Chavan, H., and Sane, S. 2011. Mobile Database Cache Replacement Policies: LRU and PPRRP. In *Advances in Computer Science and Information Technology* (pp. 523-531). Springer Berlin Heidelberg.
- [9] Kumar, A., Misra, M., and Sarje, A. K. 2006. A New Cost Function based Cache Replacement Policy for Location Dependent Data in Mobile Environment. In *The 5th Annual Inter. Research Institute Student Seminar In Computer Science, Iriss*.
- [10] Kumar, A., Misra, M., and Sarje, A. K. 2008. A predicted region based cache replacement policy for location dependent data in mobile environment. *International Journal of Communications, Network and System Sciences*, 1(1), 79-94.
- [11] Joy, P. T., and Jacob, K. P. 2012. A Comparative Study of Cache Replacement Policies in Wireless Mobile Networks. In *Advances in Computing and Information Technology* (pp. 609-619). Springer Berlin Heidelberg.
- [12] Dar, S., Franklin, M. J., Jonsson, B. T., Srivastava, D., and Tan, M. 1996. Semantic data caching and replacement. In *VLDB*, 96, 330-341.
- [13] Ren, Q., and Dunham, M. H. 2000. Using semantic caching to manage location dependent data in mobile computing. In *Proceedings of the 6th annual international conference on Mobile computing and networking* (pp. 210-221). ACM.
- [14] Lai, K. Y., Tari, Z., and Bertok, P. 2004. Mobility-aware cache replacement for users of location-dependent services. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on* (pp. 50-58). IEEE.
- [15] Dunham, M. H., and Helal, A. 1995. Mobile computing and databases: Anything new?. *Acm Sigmod Record*, 24(4), 5-9.
- [16] Lee, D. L., Lee, W. C., Xu, J., and Zheng, B. 2002. Data management in location-dependent information services: Challenges and issues. *IEEE Pervasive computing*, 3(3), 65-72.
- [17] OMNeT++ Discrete Event Simulator. 2014. OpenSim Ltd. <http://www.omnetpp.org>