Mobile Information Systems 1 (2005) 149–166
IOS Press

Semantic prefetching strategy to manage location dependent data in mobile information systems[1]

Sang-Won Kang*, MoonBae Song, KwangJin Park and Chong-Sun Hwang
Department of Computer Science and Engineering, Korea University 5-1, Anam-dong, Seongbuk-Ku, Seoul 136-701, Korea

Abstract. In mobile computing environments, location plays an important role in determining user's interests. If service providers can supply information reflecting users' semantics, overall system performance can be improved. Users can avoid queries, resulting in unnecessary network usage. A query result can also be transferred to a local device. Through our proposed semantic prefetching scheme, we make it possible to send data to mobile clients in advance. Researchers have been, for a long time, trying to develop a method to overcome mobile information system issues. These issues consist mainly of low bandwidth, high delay and frequent disconnections. In this paper, we represent the semantic prefetching strategy, which utilizes users' interests, relating to a users' current location. Our performance results show that this method is superior when compared with traditional semantic cache schemes.

Keywords: Semantic prefetching strategy, location dependent data, semantic locality, cache replacement, mobile computing

1. Introduction

Users freely move in mobile computing environments. However, as their location changes, the corresponding value of their location attribute may vary. Location-Based Services (LBSs) and Geographic Information System (GIS), are core attributes of mobile systems, both of which provide location based services. These systems recognize a users' particular location as important information, representing not only a cellular based system, but also, enabling the use of Global Positioning Systems (GPSs). Location information, with corresponding system developments, are becoming a critical issue in mobile computing environments. Consequently, the endeavor to provide a greater precision, in terms of predicting users requirements, for particular location is a continuing focus. Through this paper we contribute to this goal, by proposing a new mechanism to improve the precision in locating a particular user.

Traditional cache management schemes were first realized in a wired environment [3,10]. The cache scheme is becoming even more important in the wireless environment due to the fact that wireless technology still lacks the level of stability that mobile systems require, in terms of reliable communication. Contrary to a wired environment, which does not need to keep track of user's location, wireless

*Corresponding author: Sang-Won Kang, Department of Computer Science and Engineering, Korea University, 5-1, Anam-dong, Seongbuk-Ku, Seoul 136-701, Korea. Tel.: +82 2 924 0547; Fax: +82 2 953 0771; E-mail: swkang@disys.korea.ac.kr.

[1]A preliminary version of this paper has appeared in The Sixth International Conference on Information Integration and Web-based Applications & Services (iiWAS2004), Jakarta, Indonesia, September 27–29, 2004.

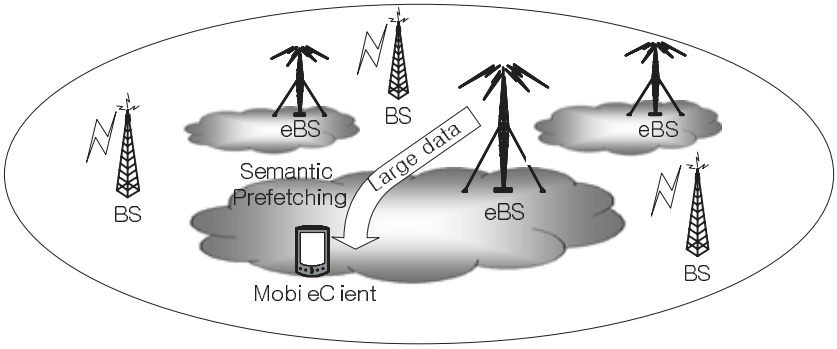


Fig. 1. Infrastructure for data.

environments possess the burden of having to constantly keep track of user's location. To define these semantics, we have applied semantic locality. We explain that characteristics of semantic locality to include not only spatial locality of an existing physical database, but also semantic information based on a users' location. Knowing the exact location of the user is essential for LBSs and GISs, and this is becoming even more important in GPSs, within cellular systems.

Semantic cache includes these localities, and represents them through a semantic description [7,9, 17,18]. Semantic cache consists of semantic descriptions and contents received from previous queries. Because the semantic cache will have a semantic description and data received from a previous query, even if the connection is lost or has entered an unreachable state, a user is able to acquire information contained in the semantic cache. In addition, traditional prefetching strategy makes up for weak points in the cache strategy. The prefetching scheme can reduce a cache miss and additional network use occurring from a cache miss, by advance transmission of data.

The proposed semantic prefetching scheme successfully overcomes the limitations of wireless environments by transferring data and semantic information to users simultaneously. The proposed scheme focuses on data services, based on the assumption that the service is supported by a connection capable of high bandwidth rapid transmission [6]. We propose the semantic prefetching scheme with a semantic description for LDD in mobile information systems.

Figure 1 shows the infrastructure for data, transmitted to specific areas. The system consists of high bandwidth fine-grained cells. Considering the philosophy of "anytime anywhere" services, a data communication system must also be designed to accommodate mobile environments.

This paper is organized as follows: In Section 2, we describe some of the recent research related to cache management schemes in mobile information systems. In Section 3, we provide a model suitable for mobile information systems. Section 4 represents a proposed semantic prefetching technique and deals with query processing and cache management schemes. In Section 5, we explain simulation results and analysis according to various scenarios, – divided into uniform and zipf distribution. Finally, in Section 6, we conclude our paper, and discuss suggestions for future work.

2. Related work

Our research approaches two aspects of data and mobility in mobile environments. We consider two components relating to a users' particular location. This is divided into sections denoted as "location-aware" and "location-based". "Location-aware" means to know where the client is located, whereas

"Location-based" means data is transmitted to user relating to the current location [1,19,20]. These terminologies both need to be applied for efficient data management in mobile environments. With these terminologies, conventional cache and semantic cache strategies have been constantly developed to form successful mobile information systems. Strategies have been introduced to reduce the network traffic, and if disconnection occurs, a cache guarantees a hit ratio using information in the client cache [3,4,10, 11]. Prefetching research has not only improved the level of data accuracy but also reducing cache miss. With this strategy, Infostation infrastructure has the ability to transmit large amounts of data in congested area [5,6].

2.1. Cache and semantic cache

Traditional cache mechanism is based on page cache. Page cache is sensitive to physical database organization. If we consider spatial locality in LBSs, the cache hit ratio can be diminished within the page cache. This results in frequent network communication for gaining the requested data since the page cache does not retain semantic information, clients largely depend on the status of the network connection with the server. We pay attention to how close query results conform to the local cache. During the disconnection period we cannot ensure consistency of a query result supporting a user's semantics. At this point in time, recent research has proposed semantic cache techniques for gaining information that includes users' interest and semantic method [4,12–15]. The semantic cache maintains semantic descriptions and contents of previous queries to the mobile client. The semantic cache transmits total data not as a page unit but as a semantic region or a semantic segment unit.

If the semantic cache is used with a main cache, we can reduce communication cost and network traffic. This is because clients only transfer remainder queries in the case of probe and remainder queries. Semantic cache strategy has overcome a drawback occurring with disconnection states in mobile computing environments, combining data and semantic descriptions. The semantic cache is composed of a set of elements referred to as semantic descriptions. We represent the semantic description $S_d = \{S_R, S_A, S_P, S_L, S_C\}$ to form [7,9,16–18]. S_R represents relation, S_A indicates attribute, S_P marks predicate with location information named to S_L. S_C has contents of data. The examples show a statement of the semantic cache query. L_x, L_y represent the user's x, y coordinates in current position, and h_{xpos}, h_{ypos}, d_{xpos}, d_{ypos} represent the location of the LDD requested.

This example shows that it reconstructs general query statements into an SQL pattern.

– Query 1: Show me the name of hotels within a radius of 10 kilometers.

  S1 : Select $Hname$ From $Hotel$
  where $(L_x - 5 \leqslant h_{xpos} \leqslant L_x + 5) \wedge (L_y - 5 \leqslant h_{ypos} \leqslant L_y + 5)$

– Query 2: Give me the names of department stores within 10 kilometers which are open between 10 a.m. to 8 p.m.

  S2 : Select $Dname, Type$ From $Dep\ Store$
  Where $(L_x - 10 \leqslant d_{xpos} \leqslant L_x + 10) \wedge (L_y - 10 \leq d_{ypos} \leqslant L_y + 10) \wedge (5 \leqslant time \leqslant 8)$

– Query 3: Give me the names of the hotels within 5 kilometers whose prices are below $50.

  S3 : Select $Hname, Room$ From $Hotel$
  Where $(L_x - 5 \leqslant h_{xpos} \leqslant L_x + 5) \wedge (L_y - 5 \leq h_{ypos} \leqslant L_y + 5) \wedge (Price < 50)$

If we are aware of correct coordinates of a particular region regarding location information, a mobile client can consider established queries based on these coordinates or the specific region. The semantic cache strategy can yield expected results using a set of elements called the semantic predicate. Research of cache replacement regarding LDD have been proposed by Franklin [4] and Dunham [7].

2.2. Prefetching

The prefetching technique reduces the cases of cache miss and cost of bandwidth usage. The prefetching technique can expect high efficiency regarding requests as it supports fast query response time.

The conventional prefetching technique is used to overcome the delay condition of processors during execution periods. It is generally divided into hardware prefetching and software prefetching [8]. The hardware prefetching does not need the assistance of a programmer or compiler. Without taking the trouble to recompile, it can dramatically improve the speed of existing programs. It does not increase the size of program, as unnecessary instructions are not added to the code base. As a result, hardware prefetching can apply to data and cache for the purpose of efficiency, whereas software prefetching is applicable for specific applications. As a software prefetching is not considered as part of the hardware, it can be inexpensive, and therefore, a benefit in reducing prefetching time.

Our paper focuses on data prefetching among several prefetching techniques, which are components of hardware prefetching. Moreover, our paper represents some methods to utilize the data prefetching in mobile environments. Our paper overcomes problems of data consistency using the semantic description.

2.3. Infostation

As the necessity of data communication becomes more important within a cellular environment, mobile information systems are making a new attempt to adjust infrastructure for data. Many researchers are processing a new challenge, which is to consider data communication and multimedia service in a cellular infrastructures.

For construction of new architecture, we must consider one of two solutions. One approach is to make an exclusive transmission path for data, remaining at an existing base station. Another approach is that the existing base station is modified, yielding an expanded structure. In this way, the data communication approach is a continuously progressing effort to accomplish new challenges, passing on the goal of sending suitable information, to users.

Infostation provides wireless information services in a limited coverage area. As users with notebook computers or PDAs pass an Infostation, they automatically receive useful information, without stopping to enter requests. This could be information that is most relevant near the Infostation, such as restaurant data, and university course information. Or it could be information of general interest, such as news, articles or music.

Research regarding Infostations are focused on data services, supporting high bandwidth availability with low transmission times [6]. These service areas guarantee stable transmission without frequent disconnections. In developing a new technology such as 3 G or 4 G, data communication services deliver multimedia data packets. Since our paper applies to the prefetching technique, a system model explains expanded base stations (eBSs) that are similar to the Infostation concept. Infostations, having a high bandwidth, – ultra high speed access(n 100 Mbps), providing data services with high bit rate and capacity at the application level. If an Infostation is used in a congested area that includes many data and users, system performance will improve. Figure 2 represents the Infostation environment.

In Fig. 2, a mobile client obtains location dependent data in an Infostation area, and the GIS server generates geographic information based on the area. The location of each mobile client is recognized by the GPS satellite. Each Infostation is connected in a wired environment. GIS DB1, GIS DB2 and GIS DB3 represent a database composing LDD about each region.

3. The expanded Base Station(eBS) Model

The proposed model is based on client-server architecture divided under an eBS area. The eBS supports data transmission using high bandwidth. Our system model assumes that mobile clients can get LDD related to a query from the server, such as in Fig. 3.

We assume a few situations for the system model. First, a mobile client has visited each eBS area at least once. The second is that each server in the eBS area retains log files left by users. Log files are written by query contents represented from semantic descriptions, and are managed by the server. A mobile client manages to cache and query processing information. Our cache replacement strategy considers that semantic descriptions and data in a client cache is efficiently managed with a semantic description and data received from the new prefetching technique. We define a region segment as composed of semantic descriptions and data, and thus these contents will be specified for mobile information systems.

The prefetching technique statistically sends the most frequently used information in the current location, but our proposed technique referred to semantic prefetching, sends semantic information based on users' log files.

Our system model assumes that the network architecture is organized into two cases. One is continuously constructed under a cellular system of one eBS per cell. The other is only constructed by a discrete architecture offering one eBS per fine-grained area. These are necessary in order for users to support a volume of informational content. The present wireless environments are continuously attempting to change the existing infrastructure for data communication, therefore this paper assumes the former case for utilization under an identical data communication model.

4. Semantic prefetching

Our scheme with semantic information could help the users obtain information relating to their interests. Although users pass by an eBS area at high speed, a semantic prefetching scheme correctly supports requested data with semantic information. We examine this semantic prefetching technique in detail.

4.1. Basic concepts

The cache contents of a mobile client must be corresponded to within a query. A server sends information to a mobile client, expected for future use with a semantic description $F_d = \{F_R, F_A, F_P, F_L, F_C\}$. F_R defines to the relation and F_A defines the attribute. F_P represents the predicate, expressing the location range of the mobile client, referencing the F_L which means prior location. Finally, the F_C are contents satisfied to locations visited previously. This paper describes cache replacement strategy and re-definition of LDD with semantic description using the AND, OR operations. It is applied to each *Semantic-Least Recently Used*(*S-LRU*) strategy and *Semantic-Least Frequently Used*(*S-LFU*) strategy.

Definition 1. Given a Log File $L_f = \{R_i\}$ and its attribute set $A = \bigcup A_{R_i}$, $1 \leqslant \mathrm{i} \leqslant \mathrm{n}$, a **Compare Predicate** of D, P, is of the form $P = a\ op\ c$, where $a \in A$, $op \in \{ \leqslant, <, \geqslant, >, = \}$, c is a constant in a specific domain, location of a, $L = \{ L_x, L_y \}$.

Semantic prefetching is composed of a set of items, defined as a prefetching description.

Definition 2. Given a Log File $L_f = \{R_i\}$ and its attribute set $A = \bigcup A_{R_i}$, $1 \leqslant \mathrm{i} \leqslant \mathrm{n}$, a **Prefetching description**, F_d, is a tuple $< F_R, F_A, F_P, F_L >$, $F_R \in D$, $F_A \subseteq A_{F_R}$, and $F_P = P_1 \vee P_2 \vee \ldots \vee P_m$.

where each P_j is a conjunctive of compare predicates, i.e. P_j=$b_{j_1} \wedge b_{j_2} \wedge \ldots \wedge b_{j_l}$, each b_{j_t} is a compare predicate and F_L=$\bigcup_{i=1}^{n} \bigcup_{j=1}^{m}$ (L_i,L_j)

Lemma 1 and 2 are formally defined by the existing semantic segment used in a semantic cache mechanism. In definition 3 and 4, we assume that the user selects the function θ for the purpose of comparing prefetching descriptions to semantic descriptions.

Lemma 1. A **Semantic description**, S_d, is a set of semantic segment, S_d=<S_R, S_A, S_P, S_L>

Lemma 2. A **Query** Q is a semantic segment, <Q_R, Q_A, Q_P, Q_L>

Definition 3. F_{d_i} is a prefetching description. S_{d_j} is a semantic description. Let S, T and W be sets with semantic descriptions

$$S = \{F_{d_i} \mid 0 \leqslant i \leqslant n\}, T = \{S_{d_j} \mid 0 \leqslant j \leqslant n\}, W = S \cap T.$$

Definition 4. A semantic description table $\mathcal{T}$ is defined as

$$\mathcal{T} = \{\theta, \mathcal{W}, (\mathcal{S} \cup \mathcal{T}) - \mathcal{W}\},$$

where θ is a sorting function. If we suppose that S-LFU follows the frequency of the query divided into two parts: hot data and cold data, by users' semantic description. In S-LFU, the factor of θ is frequency. Whereas, we suppose that S-LRU follows the recent access of the query: recent data and old data by users' semantic description. In S-LRU, the factor of θ is time.

In this situation, LDD has an effect on F_L and S_L. F_L are previously physical coordinates queried from the current area, which is maintained to the server. Also, S_L are physical coordinates saved to the cache. Cache must be renewed to the current location. In addition, when a query part of a cache accords with the query part of F_L received from an eBS, the cache will maintain its probe query, the remainder query is again transferred to the server through the general BS. As a mobile client can go into a new eBS area in several directions, results of the request vary according to the location. When a mobile client enters a new area, how the client cache is organized is important. First of all, the common part, referred to as part of the unified contents, and its remainder part are organized into a set table maintained by five semantic description elements.

In this paper, our scheme manages LDD maintained in server and client cache. Data maintained in the cache must conform to user queries. This is satisfied with a semantic description table. The data and semantic description can make one part a common set and the other part an ordering set. Ordering sets choose between the order of query frequency and the order of the latest query. Therefore, although a mobile client moves in a eBS area, the data and semantic description in the cache keeps up their validation.

4.2. Query processing

A segment of LDD in a query maintains a semantic description and data content. After making a new segment part under continuous queries like Fig. 4, the semantic segment is disjoined to probe queries and remainder queries. Probe queries are parts accorded with the semantic description table reconstructed by semantic prefetching. Concurrently, remainder queries are not accorded with the semantic description table. If the semantic segment includes well-analyzed query information, remainder queries will decrease considerably.

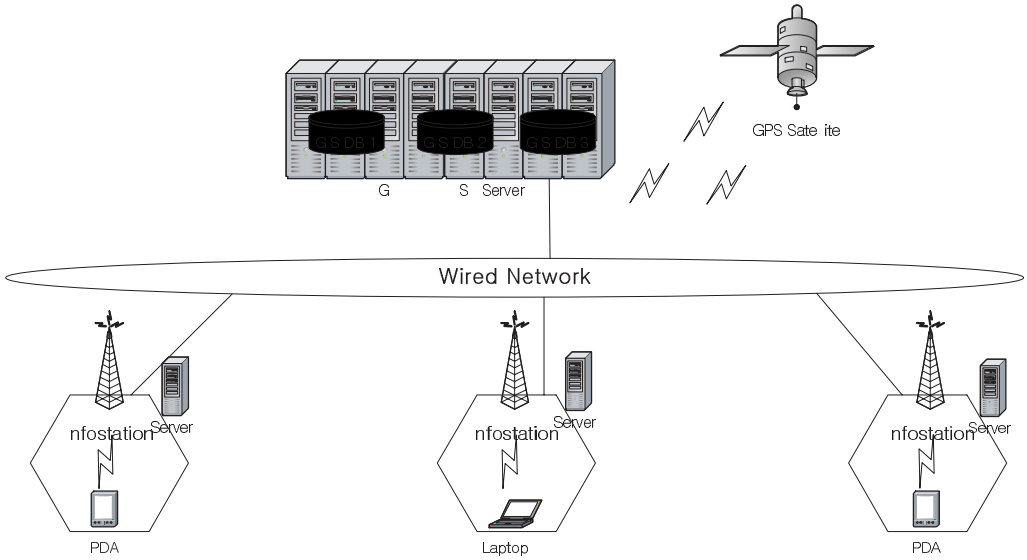


Fig. 2. Infostation architecture.

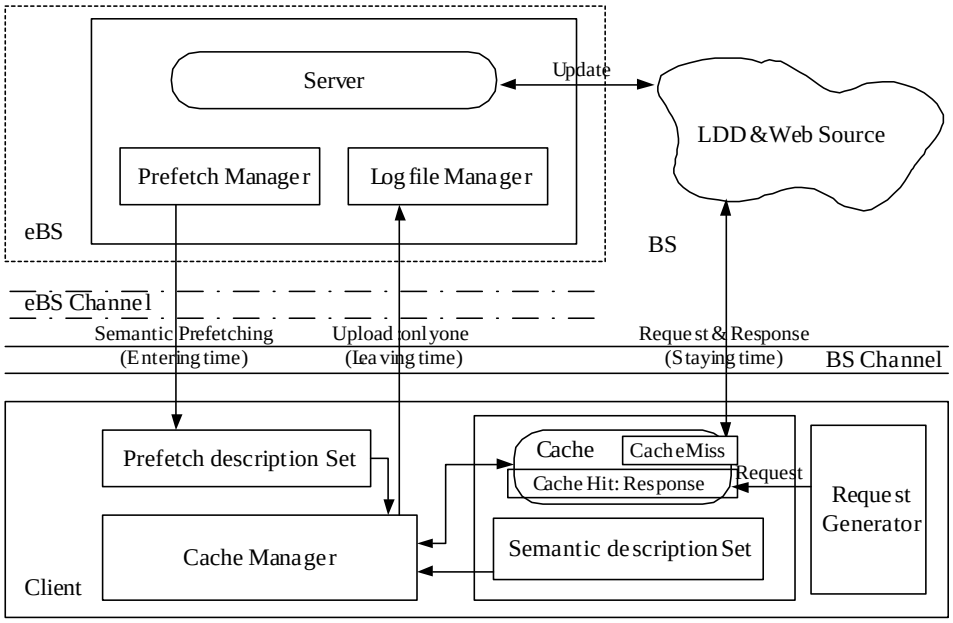


Fig. 3. eBS model.

Consequently, we can get responses regarding requested queries using probe & remainder queries. The semantic cache strategy has increased the hit ratio within a mobile client cache in an existing network architecture, but if a user in a fine-grained area wants to get information, the semantic cache strategy makes it impossible to support large-scale contents of LDD. Semantic prefetching strategy overcomes these limitations. Minimizing the contents of remainder queries, semantic prefetching strategy reduces additional communications cost and query response time.

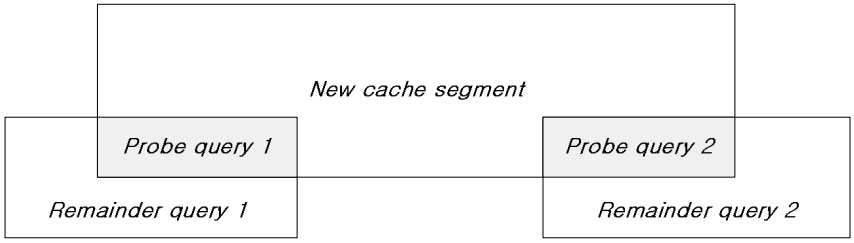


Fig. 4. Probe & Remainder query.

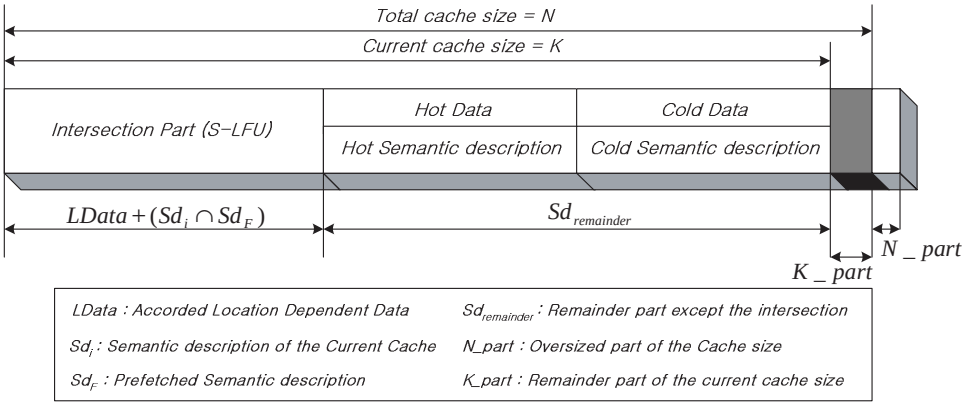


Fig. 5. S-LFU Cache Management.

4.3. Cache management scheme

When caching the received data, we would also store the received semantic description in the cache. After inserting the received semantic description into the existing cache contents, under the S-LRU strategy, the cache can be reconstructed in an intersection and union part. The former is a correspondence part and the latter includes intersection part and all parts of both the semantic descriptions. A remaining part of the cache can be maintained.

In this paper, including semantic information, cache management schemes have two different kinds of techniques such as an S-LRU strategy, which is least managed by recently used data, and an S-LFU strategy, which is least managed by frequently used data. Our simulation performs the S-LRU strategy for several experiments.

The S-LFU strategy will deal with statistical data related to the user's frequency after the implementation. If the cache data of a mobile client fully contains a user's query, the user can immediately get data from the local cache. The cache size of a mobile client is limited, so it is important to manage several data streams received from server, using a filtration like replacement strategy. In the S-LFU strategy, hot data and semantic descriptions are represented in the order of high frequency information. On the contrary, cold data and semantic descriptions are represented to the order of low frequency information. The server maintains both hot and cold information by statistically calculating the frequency of contents. In this case, the server can calculate the frequency by counters and information can be organized for a mobile client cache, considering the order of frequency, when a mobile client requests information. Figure 5 represents the S-LFU strategy. It shows the most basic cache management of the S-LFU strategy.

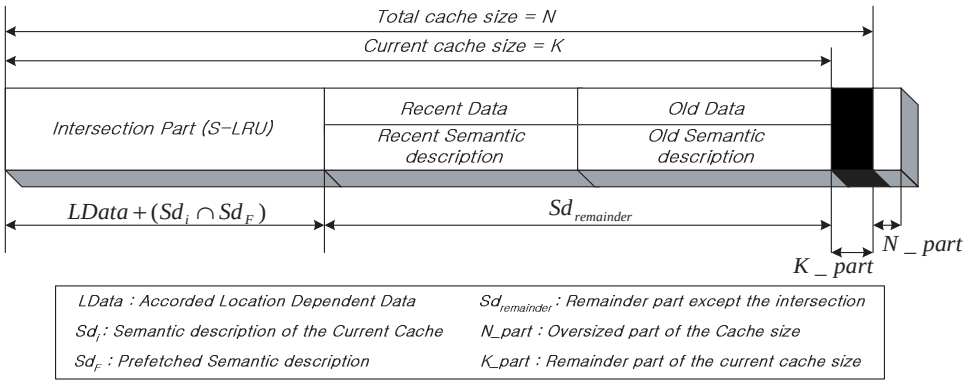


Fig. 6. S-LRU Cache Management.

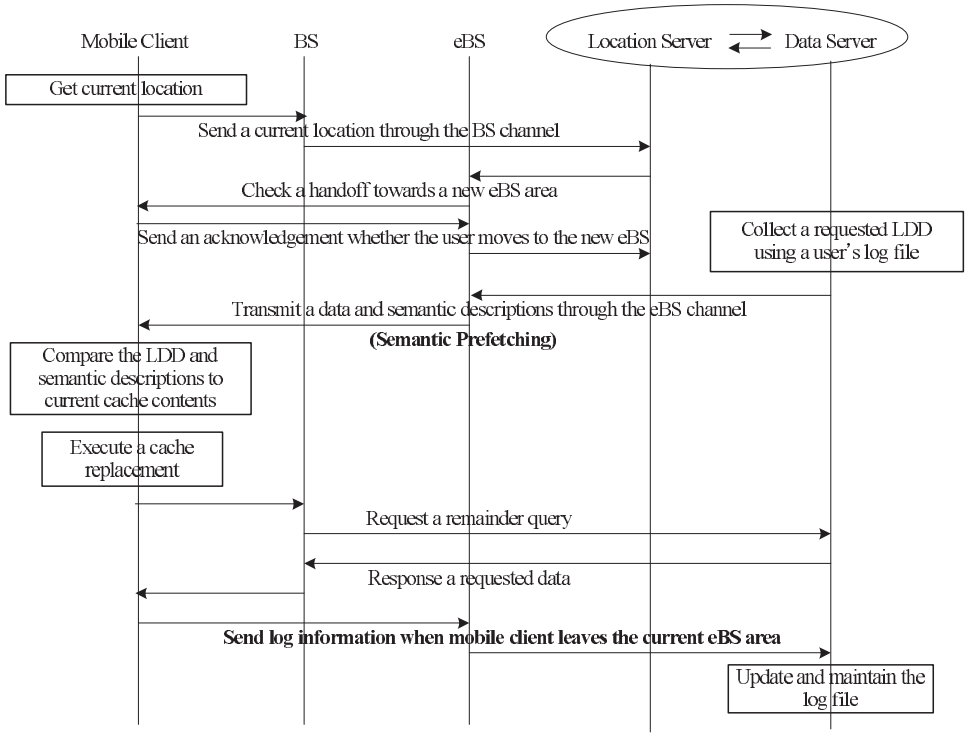


Fig. 7. Semantic prefetching based on semantic cache.

The determination of how to consider the order of frequency is adjusted in a user's environment during implementation.

We propose the S-LRU strategy for managing our cache like Fig. 6. In Figs 5 and 6, the total cache size defines *N* and the current cache size can represent the *K/N %* notation by the arbitrary *K* value, and intersection part also denotes the *LData*+($Sd_c \cap Sd_F$). In the current cache size, the remainder part, not including the intersection part denotes $Sd_{\text{remainder}}$. The remainder part of the current cache size defines the $K\ part$. That is, we can represent to the $(1 - K/N)\%$ notation equal to the $K\ part$. In Figs 5 and

```
Mobile : mobile client
LData_F : LDD predeterminated to prefetch
S_dF : semantic description predeterminated to prefetch
Log_file : remains the information about the data used by users
M_data : element of data used by mobile client about the query
M_sd : element of semantic description used by mobile client about the query

LData_F = EMPTY
Function Find_data
 if (Mobile.in=1) // When the mobile client enter into the eBS area
  then Read Log_file
       Select M_data, M_sd
            LData_F = M_data;
            Sd_F = M_sd;
 else if (Mobile.in=0) //When the mobile client leave the eBS area
       Collect M_data, M_sd
       Write Log_file //Record to the database
       Save Log_file
       //Periodically store to the database according to the updating
Function Send_data
  LData_F = FULL, S_dF = FULL;
  Send LData_F, S_dF
```

Fig. 8. Sertver.

6, the contents of the cache are maintained in the data and semantic description. We denote the $N\ part$, which is oversized to the cache size N. If a mobile client receives the excessive data under the semantic prefetching scheme, the cache deletes the contents of $N\ part$.

The relation between a server and a mobile client is shown to Fig. 7. According to the progressing step of Fig. 7, although the eBS network has a disconnection status, the mobile client can get the result for a requested query from the updated cache. The server maintains the semantic description part and sends a users' updated information made by a former semantic description, to a client when the client enters the area. In this case, overhead occurs in the server, since it has to maintain several users' information. This paper has no consideration regarding a server's overhead.

Execution functions of server and mobile client show the server and mobile client algorithm. This is represented to each execution part.

Example 1. supposes a mobile client moves around the eBS area. The server side action is determined from a mobile client's location. When a user enters an eBS area, the server transmits predicted information in advance. For efficient transmission, the server collects data based on previous information and semantic predicate. This is the main contribution of ideas called to semantic prefetching. The ability of the server depends on data prediction.

In the server algorithm, there are two functions, $Find\ data$ and $Send\ data$, each represented to prefetch requesting data. The $Find\ data$ function identifies the fact that the location of a client changes from one eBS to another using the GPS, updated data, and other necessary data is found using information organized in log files. Furthermore, this function maintains log files in the database when a mobile client leaves this area and periodically updates data. After finishing the $Find\ data$ function, the $Send\ data$ function sends the requested data to the client. The server transfers prefetching contents from the client using the $Send\ data$ function.

Example 2. Suppose that a mobile client exists in an eBS area. Immediately on entering the eBS area, a mobile client receives semantic prefetching data. Although the mobile client keeps a cache, a user

$LData_i$: LDD remained to the current cache
S_{d_i} : semantic description remained to the current cache
$LData_c$: final LDD organized to the cache
S_{d_c} : final semantic description organized to the cache
$S_{d_{remainder}}$: the discorded remainder semantic description
N_part : a oversized part to the total cache size
(Current cache : K, Total cache : N)
$LData_i + S_{d_i}$: contents of the current cache itself
$LData_F + S_{d_F}$: contents received to the server

$LData_c$, S_{d_c} = EMPTY
Function Receive_data
  add $LData_F$ to $LData_i$
  add S_{d_F} to S_{d_i}
Function Compare_data(S_{d_F}, S_{d_i})
 if ($S_{d_F} \cap S_{d_i}$) // When the mobile client enter into the eBS area
  then $S_{d_c} = \{S_{d_F} \cap S_{d_i}\}$
        $S_{d_{remainder}} = (\{S_{d_F} \cup S_{d_i}\}$ - $\{S_{d_F} \cap S_{d_i}\})$
          add $S_{d_{remainder}}$ to cache using the order of recent access
        $S_{d_c} = S_{d_{remainder}}$
 else if
        add S_{d_F}, S_{d_i} in recent order according to the query access
Function Select_data ($LData_i$, $LData_F$)
  $LData_c = LData_i$;
  $LData_c = LData_F$;
Function Delete_data
 if ($LData_c + S_{d_c} \leq$ N)
  case 1 : ($Mobile = 1$)
      save $LData_c$, S_{d_c} to the cache
  case 2 : ($Mobile = 0$)
      upload $LData_c$, S_{d_c} to the Log_file
 else if
  delete $N\_part(S_{d_c}, LData_c)$

Fig. 9. Mobile client.

is limited in gaining a cache hit. We consider a new cache replacement strategy for using semantic prefetching. Mobile clients select cache contents and delete useless data, depending on the cache size.

In a mobile client algorithm, there are four functions, represented by $Receive\ data$, $Compare\ data$, $Select\ data$, and $Delete\ data$. The $Receive\ data$ function adds the contents of prefetching data to the current cache. The $Compare\ data$ function remakes the contents of the cache by comparing contents received by semantic prefetching the contents of the current cache. The $Select\ data$ function maintains contents of the new client cache. Finally, the $Delete\ data$ function determines whether to delete after checking the cache size.

5. Performance evaluation

We evaluate the performance of our model through extensive simulations using scenario data. We describe our simulation environment in Sections 5.1, 5.2, 5.3 and 5.4. We demonstrate our experiment results in Section 5.5.

5.1. Resource and model parameter

The cache strategy experiments using semantic cache and semantic prefetching consider the total cache hit. The factor sets each value of parameters. It is compared with the cache hit ratio regarding the

Table 1
Parameter setting

Parameter	Description	Setting
CACHE MAX SIZE	Size of client cache	10% of server
S CACHE MAX SIZE	Size of server cache	100
TOTAL QUERY NUM	No. of query	400
QUERY NUM PER CELL	No. of query per cell area	10
NUM CELL eBS	No. of eBS area	5 by 5
X Y eBS	Physical coordinates of x, y about 5 by 5 eBS area (x,y)	100 by 100
NUM eBS	No. of eBS	25
NUM LDD	No. of distributed LDD	400
TYPE LDD	No. of data type	30
QUERY SEGMENT SIZE	Size of query segment	20, 30

Table 2
Scenario

Scenario	A	B	C	D
Distribution	Uniform & Zipf	Uniform & Zipf	Uniform & Zipf	Uniform & Zipf
Mobility	Low	High	Low	High
Query pattern	Various	Similar	Similar	Various

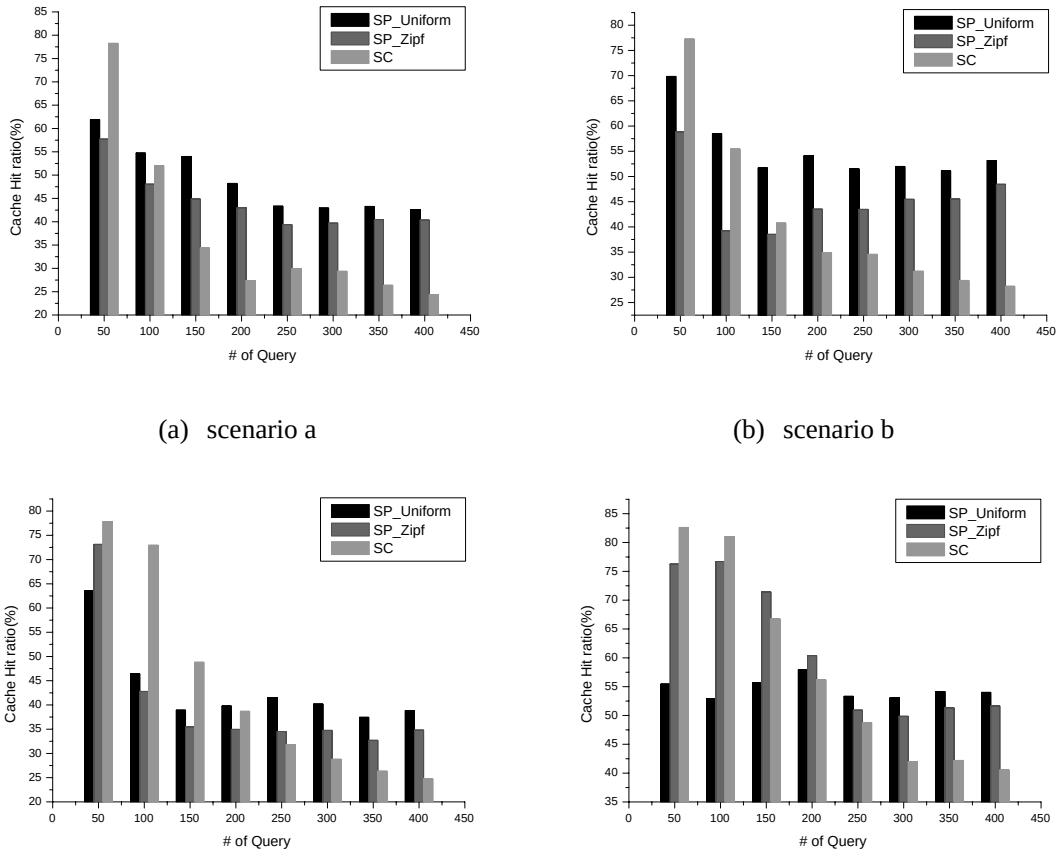
number of queries, and query segment size. For this simulation, we basically coordinate one server per eBS. Table 1 represents parameters used to our simulation model. $CACHE\ MAX\ SIZE$ sets from the 10% of the $S\ CACHE\ MAX\ SIZE$. After querying, the mobile device sequentially checks the contents of the cache for responding to the query.

In this paper, the replacement strategy regarding cache hits utilize the S-LRU strategy. Having specific bandwidth, we suppose that our network utilizes the FIFO queue model. We suppose that networks for the eBS and the BS both exist in our model together. Using 5*5 service area for experiments, the total number of queries requested by mobile client is 400. We suppose that the number of continuous queries per area is 10. The number of cache hits regarding the probe query differs depending on the size of the query segment. In our simulation, we have several experiments with various sizes of query segments. A total of 30 pieces of data is used in our simulation. We implement 400 pieces of LDD, including each data type, set into the service area fixed to the eBS. Our eBS area are divided into physical 100*100 coordinates. Using the coordinates, we represent the location of mobile client and LDD.

During this papers research, consideration related to the update is out of the question, as the overhead pertaining to update operations, doesn't apply. Query information and previous data used by a user is logged to the server. When a client re-enters the service area, the user can receive information. The server must support the service and database of users. For a semantic cache and semantic prefetching strategy, semantic regions are maintained through buffer management for the user. In semantic cache and semantic prefetching techniques, we use semantic information for supporting the proposed scheme, no consideration is given for an overhead of indexing management for mobile clients. Parameter values used to compare and analyze our simulation are determined depending on the number of user's queries, the size of the query segment between the normal semantic cache and semantic prefetching semantic cache.

5.2. Simulator for semantic prefetching technique

Our simulator used for verification is implemented in the Java language. We simulate and compile a model using a Sun Blade 1000. Experimental environments are of client-server architecture, used for



(a) scenario a (b) scenario b

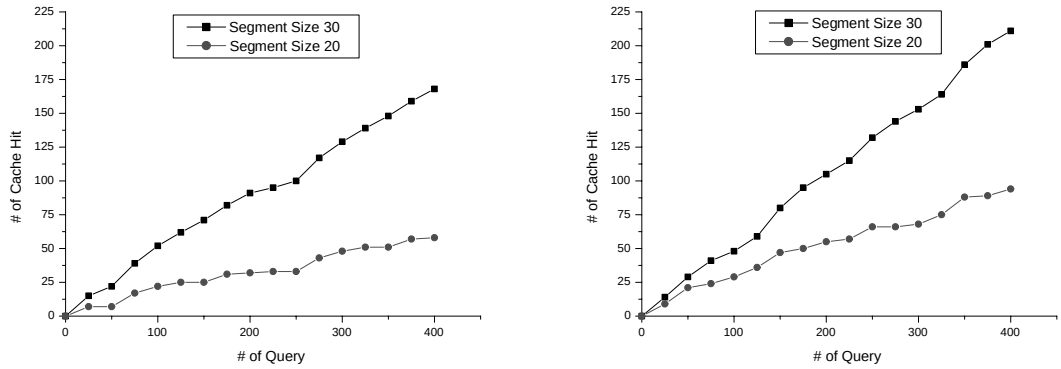
(c) scenario c (d) scenario d

Fig. 10. Comparison of each distribution.

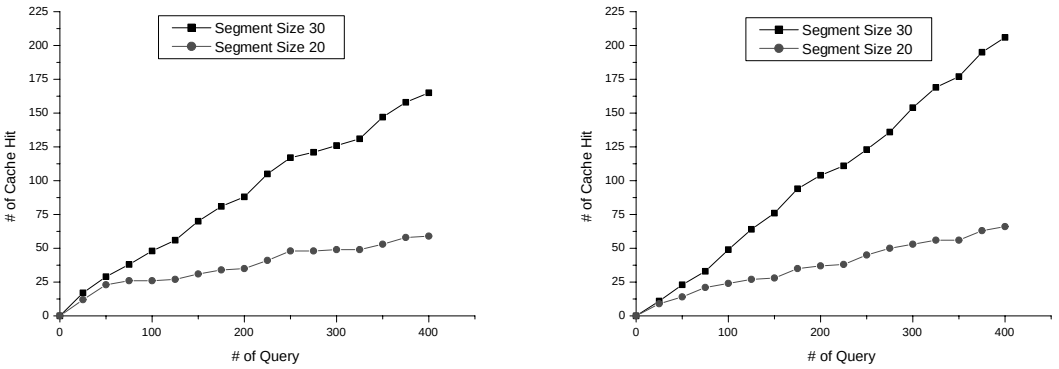
mobile communication. The simulator is made of an eBS class and a client class, applying methods of the client-server architecture. An eBS class includes a method to manage data made from an experiment of uniform distribution and zipf distribution, and client class represents the cache with current location coordinates for clients, data attributes, and segment size together. We suppose that the client has visited the area at least once. Therefore, at first, users can receive the data using previous information. Data used to our simulation is prefetched with each distribution method, and query patterns sent from the client is organized differently by moving patterns according to the 4 scenarios.

5.3. Simulation assumption and specification

In existing prefetching, the server provides data, frequently used by users, to the client without semantic information. Considering the frequency of data, semantic prefetching provides previous query information and services maintained by semantic information about a user. In this paper, when clients move around a set location, and if characteristics consist of continuous queries, we define semantic relation and semantic attributes as one set. If a user queries different attributes of the same relation, we assume that it is a different data type. The data gained from our simulation is LDD. The semantic



(a) scenario a (b) scenario b



(c) scenario c (d) scenario d

Fig. 11. Comparison of segment size.

predicate including the condition attribute, is organized by the segment area, having a specific distance from the current point. When the query is processed, the queried area is overlapped with a nearby cell area, we define the query part as assigned to the physical address for the relevant area. We repeatedly organize the moving pattern by visiting the 20 areas.

5.4. Scenario

In the processing part of our simulation, we use 4 scenarios for presentation of real-world moving patterns. Each scenario determines 30 data types and physical locations of X, Y coordinates of each data represent the same data type using the uniform or zipf distribution. A uniform distribution, sometimes also known as a rectangular distribution, is a distribution that has constant probability. On the other hand, zipf distribution means a distribution of probabilities of occurrence that follows Zipf's law.

In these 4 scenarios, it is important to determine the moving pattern of clients. We can use a random walk model for real-world moving patterns, but we set specific moving characteristics of the client for the simulation as follows. Examining 4 scenarios, we consider the different mobility and query patterns. As

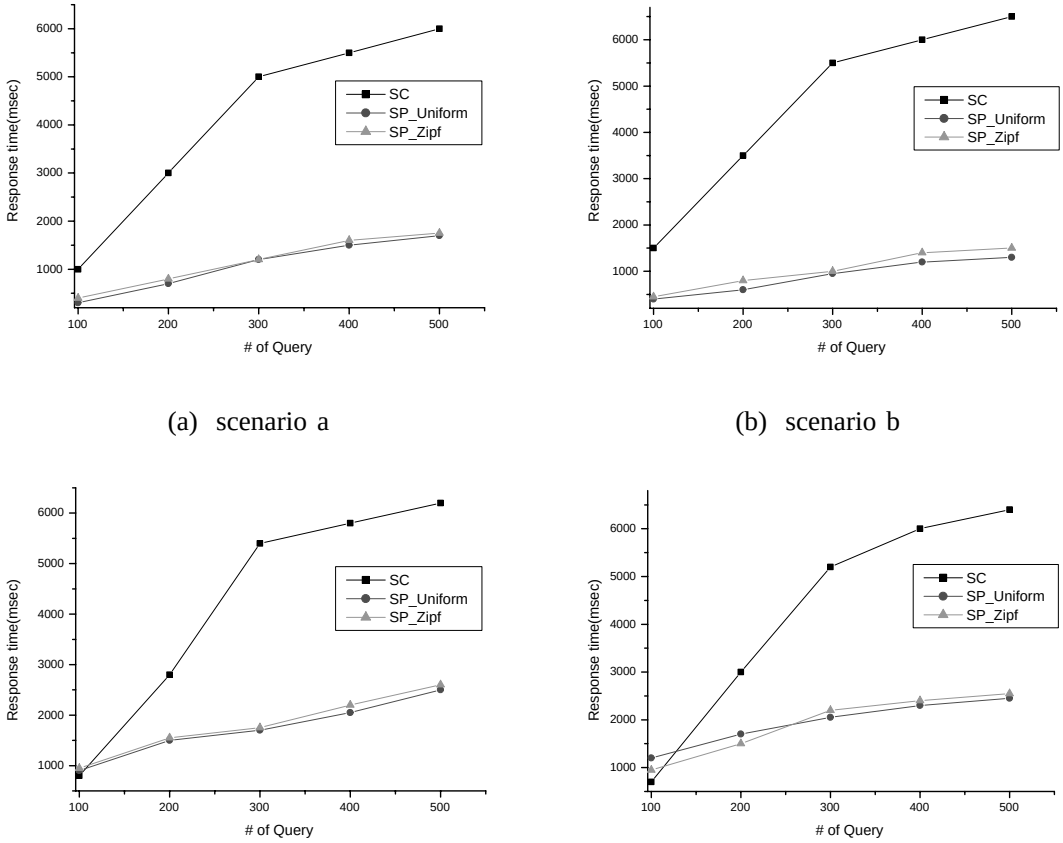


Fig. 12. Comparison of response time.

we discuss the scenarios, we compare traditional semantic cache to prefetching semantic cache, applying uniform and zipf distribution.

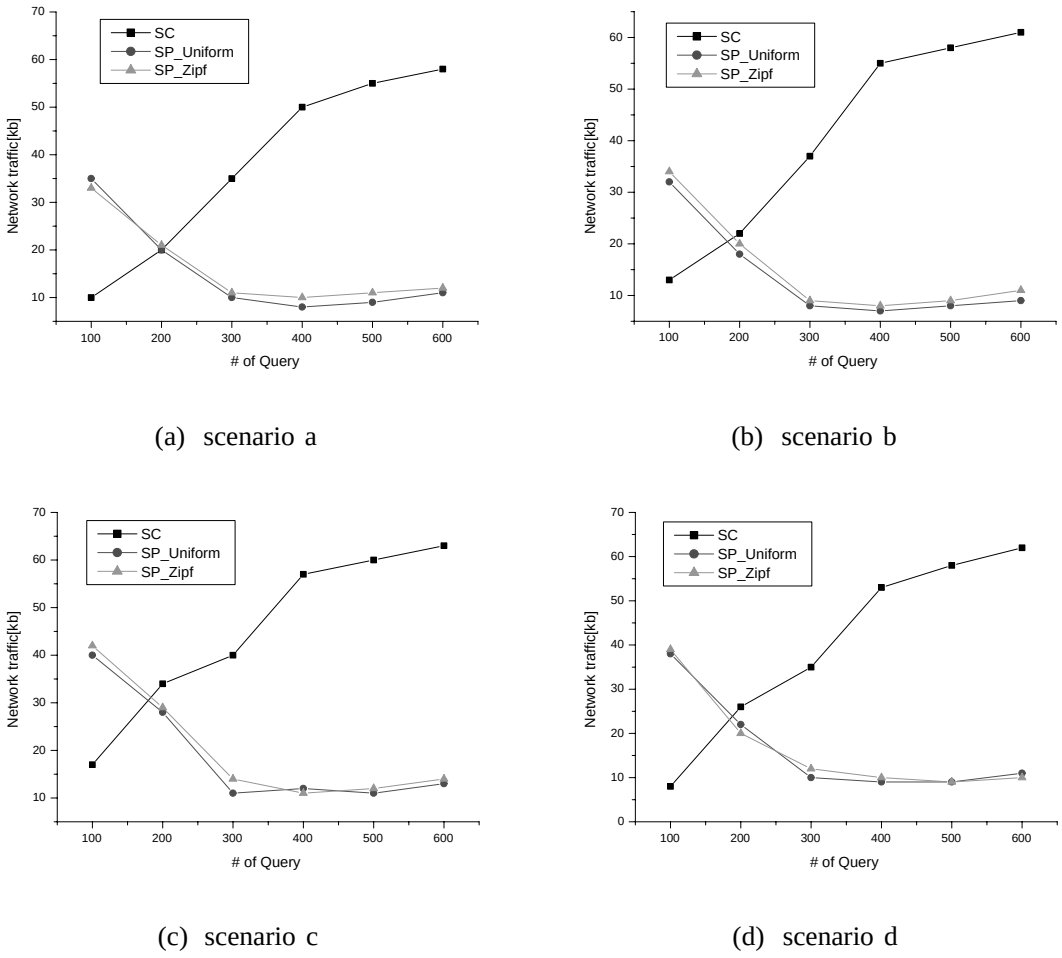
5.5. Experiment results

As shown to Fig. 10, the point represents the cache hit ratio using the semantic prefetching strategy is higher than the traditional semantic cache. We try to adjust factors such as proper cache hit, moving pattern, and query pattern. We compare traditional semantic cache to semantic prefetching using uniform distribution and zipf distribution, depicted in Fig. 10.

As started above, we adopt several situations: the cache hit ratio of data requested by mobile client, determination of different or similar query pattern, definition of low or high locality, space of the remaining local cache and re-visiting number of the area by client.

The size of a segment depends on the request type of a user. In this experiment we assume that the query pattern of a user is adequate and regularly adjusts the size of a segment about each query.

Figure 11 presents a comparison of the number of cache hits, adjusting the query area, that is, the size of a segment according to each scenario. The size of a segment is determined as the size of a regular



(a) scenario a (b) scenario b

(c) scenario c (d) scenario d

Fig. 13. Comparison of network traffic.

quadrilateral made by the radius parameter, setting the current location of the user. If the parameter sets a large value, the probability of cache hits relating to the probe query increases dramatically. In this Fig. 11, each scenario shows that the number of cache hits is remarkably differentiated, depending on the size of segment. In each scenario, the rate of increase greatly reduces the case of the small segment size. Our paper organizes the size the segment by the distance when simulation executes in a proposed modeling. Therefore, we can get some limited results like Fig. 11.

We can organize the segment area including the additional elements in the real-world implementation. We consider the additional elements including the geographic character, the process of query information and the dynamic organization of the segment area. In this way, we can understand the improvement of the cache hit relating to the request of the user and how the segment area organizes.

Figure 12 represents the comparison of response time in 4 scenarios. When mobile client requests LDD, the cache plays an important part in rapidly searching data. Semantic prefetching helps the cache support data by providing a rapid response time. Compared with the semantic cache, semantic prefetching graphs show the decreasing response time. Response time is a measure that assures efficiency of semantic prefetching. Distribution of data such as uniform and zipf applies to compare semantic prefetching to

semantic cache.

Finally, Fig. 13 shows the number of bytes sent across the network per query. In this case, the use of semantic prefetching results in a significant reduction in message volume. In a network constrained environment, such communication costs may be the dominant factor. Although semantic prefetching has network traffic in the beginning, the mobile client subsequently compiles information in the local cache.

It should be noted that in environments where data communication costs are expensive, the ability to process independently of the server can result in significant savings in addition to performance gains.

6. Conclusion and future work

In this paper, we propose the semantic prefetching scheme and new cache replacement strategy. By applying the semantic prefetching through an eBS in mobile information systems, we reduce the network traffic. The server keeps semantic information represented by a semantic description and collects data in a query segment area. Although users try to request current location data, we can get LDD from the local cache without additional network connection. With respect to communications cost and availability of the local cache, we have excellent results for mobile information systems. In future work, we will develop semantic prefetching schemes for real-world applications and try various probability based approaches.

References

[1] M.H. Dunham and V. Kumar, Location dependent data and its management in mobile databases, *Proc. of DEXA'98*, 1998, 414–419.

[2] D.J. Goodman, J. Borras, N.B. Mandayam and R.D. Yates, INFOSTATIONS: A new system model for data and messaging services, *Proc. of IEEE VTC '97* **2** (1997), 969–973.

[3] A.M. Keller and J. Basu, A predicate-based caching scheme for client-server database architectures, *The VLDB Journal* **5**(2) (1996), 35–47.

[4] S. Dar, M.J. Franklin, B.T. Jonsson, D. Srivatava and M. Tan, *Semantic data caching and replacement*, Proc. of the VLDB Conf., 1996, 330–341.

[5] George Liu, *Exploitation of location-dependent caching and prefetching techniques for supporting mobile computing and communications*, Proc. of the Sixth Int. Conf. on Wireless Communications, 1994, 11–13.

[6] A.L. Iacono and C. Rose, INFOSTATIONS: new perspectives on wireless data networks, *Technical Report, WINLAB, Rutgers, The State University of New Jersey*, 2001, 1–60.

[7] Q. Ren and M.H. Dunham, Using semantic caching to manage location dependent data in mobile computing, *Proc. of MobiCom'00* (2000), 210–221.

[8] N. Oren, A Survey of prefetching techniques, *TR-Wits-CS-2000-10* (July, 2000).

[9] Luo Li, K.R. Birgitta, N. Pissinou and K. Makki, Strategies for Semantic Caching, *DEXA'01* (2001), 284–298.

[10] Daniel Barbara, *Sleepers and Workaholics: Caching Strategies in Mobile Environments*, ACM SIGMOD international conference on Management of data, 1994, 1–12.

[11] K.L. Wu, P.S. Yu and M.S. Chen, Energy-Efficient Caching for Wireless Mobile Computing, *ICDE'96* (1996), 336–343.

[12] B. Chidlovskii, C. Roncancio and M.-L. Schneider, *Semantic cache mechanism for heterogeneous web querying*, Proc. 8th Intl. WWW Conf, Toronto, Canada, May 1998.

[13] Y. Ishikawa and H. Kitagawa, *A Semantic Caching Method Based on Linear Constraints*, International Symposium on DANTE'99, Kyoto, Japan, November 28–30, 1999.

[14] B. Chidlovskii and U.M. Borghoff, Semantic Caching of Web Queries, *The VLDB Journal* **9**(2) (2000), 2–17.

[15] P. Godfrey and J. Gryz, Semantic Query Caching for heterogeneous Databases, *Proc. KRDB* (1997).

[16] Q. Ren and M.H. Dunham, Semantic Caching and Query Processing, *Southern Methodist University, TR-98-CSE-04* (1998).

[17] C.K. Ken lee, H.V. Leong and Antonio Si, Semantic Query Caching in a Mobile Environment, *MC2R* **3**(2) (April, 1999).

[18] Q. Ren and M. Dunham, Using clustering for effective management of a semantic cache in mobile computing, *In Proceedings of the International Workshop of MobiDE, Seattle, WA*, August, 1999, 94–101.

[19] H.W.P. Beadle, B. Harper, G.Q. Maguire and J. Judge, Location Aware Mobile Computing, *Proc. IEEE/IEE ICTi97, Melbourne*, April, 1997, 1319–1324.
[20] A.Y. Seydim, M.H. Dunham and V. Kumar, Location Dependent Query Processing, *Proc. of Second ACM international workshop on Data engineering for wireless and mobile access, Santa Barbara, California, United States*, 2001, 47–53.
[21] MoonBae Song, Sang-Won Kang, and KwangJin Park, On the Design of Energy-efficient Location Tracking Mechanism in Location-aware Computing, *Mobile Information Systems Journal* **1**(2) (2005).

Sang-Won Kang received his B.S. degree in Computer Science and his M.S. degree in Computer Science and Engineering from Korea University, Seoul, Korea in 1998 and 2003, respectively. He is currently a Ph.D. candidate in Computer Science and Engineering and a researcher in the Research Institute of Computer Information and Communication, Korea University, Korea. His academic interests include Mobile Computing Systems, Mobile Data Management, Location Dependent Data, Semantic Prefetching & caching, Sensor Data Networks, Indexing & Query Processing for Moving Objects.
E-mail: swkang@disys.korea.ac.kr

MoonBae Song received his B.S. degree in computer science from Kunsan National University in 1996, and his M.S. degree in computer science from Soongsil University in 1998. He received his Ph.D. in computer science from Korea University in 2005. He is a researcher in the Research Institute of Computer Information and Communication at Korea University. His research interests include moving object databases, location-aware services, context-awareness, and data management for mobile computing.
E-mail: mbsong@disys.korea.ac.kr

KwangJin Park received his B.S. and M.S. degree in computer science from Korea University, Korea in 2000 and 2002, respectively. He is currently a Ph.D. candidate in Computer Science and Engineering and a researcher in the Research Institute of Computer Information and Communication, Korea University, Korea. His research interests include location-dependent information systems, mobile databases, and mobile computing systems.
E-mail: kjpark@disys.korea.ac.kr

Chong-Sun Hwang received his M.S. degree in Mathematics from Korea University, Seoul, Korea in 1970, and Ph.D. degree in statistics and Computer Science from University of Georgia in 1978. From 1978 to 1980, he was an Associate Professor at South Carolina Lander State University. He is currently a Full Professor in the Department of Computer Science and Engineering at Korea University, Seoul, Korea. Since 1995, he has been a Dean in the Graduate School of Computer Science and Technology at Korea University. His research interests include distributed systems, distributed algorithms, and mobile computing systems.
E-mail: hwang@disys.korea.ac.kr