European Journal of Scientific Research
ISSN 1450-216X Vol.28 No.3 (2009), pp.478-491

http://www.eurojournals.com/ejsr.htm

Data Prefetching Algorithm in Mobile Environments

El Garouani Said
Faculty of Science Dhar Mehraz, Mathematics and Computer Science Department Fez, Morocco
E-mail: saidelgarouani@yahoo.fr
Tel: +212 (6) 70044305

El Beqqali Omar
Faculty of Science Dhar Mehraz, Mathematics and Computer Science Department Fez, Morocco
E-mail: omarelbeqqali@gmail.com

Laurini Robert
LIRIS Laboratory, INSA-LYON University of Lyon, Bât B. Pascal 69621 Villeurbanne Cedex France
E-mail: robert.laurini@insa-lyon.fr

Abstract

Since the beginning of the years 2000, one attends a huge spreading of mobile systems. One of the most important problems is the efficient access to data. A proposed solution to this problem is the prefetching technique which consists in putting in reserve the information before the users need it. This information is stored in the user's mobile device cache (PDA, handheld device, portable PC, etc.). This technique takes into account the user's location and allows to send the relevant data and to reject the non-relevant ones. The main goal is to allow the users to have access permanently -in the next displacement- to available information in an almost immediate way. Besides, the data prefetching permits it to improve the performances of the cache's use and allows an adaptation of the mobile devices having limited resources, and also to take into account the prediction of the possible and the most relevant way how users will request information. This approach is considered to deal with these problems and to improve response time and reduce the amount of data to mobile devices users. This work concerns a new technique called « directed prefetching », particularly adapted to information available for location-dependent mobile systems managing large volume of data, the considered study case is mobile tourism for city of Fez.

Keywords: Cache, mobile system, prefetching, optimization, XML database, GIS, pervasive system.

1. Introduction

The generalization of the wireless networks combined with the increase of the extra-light devices transforms in-depth the data-processing applications, as well in their design as in their use. The

hardware and software infrastructure is already available or about to be. First, the wireless Internet access is now operational, by the 802.11 standard (Wireless Ethernet) or by cell phone. In addition, the extra-light devices increase in number and become connected to the network: personal assistants, cell phones, smart cards, house automations systems, car or plane embarked systems, etc. All offer calculation capacities, storage or interaction in constant progress according to Bernard, Ben-Othman, and Bouganim (2001). The mobile users can reach data and carry out treatments anywhere, any time and starting from any device. Our objective is thus to accompany this evolution in the information technology field by proposing techniques to optimize information accesses. In this paper we propose a prefetching technique which takes account of the user site to send him the relevant data and reject the not relevant data. It also makes it possible to take into account:

- Treatment in the context where several users want to reach the same information at the same time.
- Profile of each user (position, speed, direction, membership or not) that is an important parameter for this technique, because it takes account of the number of requests at the same time.

In that follows, we present a prefetching technique, the basic requirements formulation for this technique in systems dependent on the user site, and algorithm.

In this paper we present a survey of relevant data prefetching and cache invalidation. We formulate some basic requirements for successful prefetching policies for mobile environments; we give a description of our prefetching technique, and we give solutions to several problems related to its implementation. After that, we present our simulation model, the simulation prototype and some experimental results.

2. Data Prefetching and Cache Invalidation

Prefetching is not at all a new concept. Since the very early days of microcomputer technology caching and soon thereafter prefetching or preloading were integral parts of processors and file systems. In this chapter a short overview over the evolution of prefetching is given. The history starts with simple read-ahead in uniprocessor and continues until the current time where prefetching is used in a variety of fields: World Wide Web, multimedia applications and mobile computing to name a few.

2.1. Prefetching in Location-Dependent Systems

In the more specific field of location-dependent systems and mobile environments, Kubach and Rothermel (2001) propose a prefetching mechanism for location-dependent systems based on infostations Badrinath, Imielinsky, Frankiel, and Goodman, D., (1996). When the user is near an infostation the information related to its area is hoarded. An access probability table is maintained where each data item is associated with the average probability that it will be requested. Only a fixed number of the first data items with the highest probabilities are hoarded for the purpose of not wasting bandwidth and the user's device resources of Zhou, Feng, and Li, (2003). De Nitto, Grassi, and Morlupi, (1998) propose a model to evaluate the effectiveness of hoarding strategies for context aware systems, based on cost measures. They apply these measures to some idealized motion models. For the two-dimensional model, the area is divided in adjacent polygons. The distance between two polygons is the number of polygons that must be traversed to pass from the first polygon to the other one. The ring k is defined as the set of all polygons whose distance from a given polygon is equal to k. All the information associated to rings 0, 1, …, k around the starting position is hoarded. The next prefetch does not occur until the user enters a polygon outside the circle k. of Park, Kim, and Cho, (2004) prefetching schemes. The first scheme uses a ring as a prefetching zone as given by the previous discussed work of De Nitto, Grassi, Morlupi, (1998). The second scheme restricts the prefetching zone to a rectangle. The last one uses the history of user movement and counts the number of visits to each place. This scheme assumes that more a place has been visited by a user, more it will be revisited by

the same user and thus restricts the prefetching zone from a rectangle to the union of places of this latter for which the number of visits is not under a threshold.

2.2. The History and Location-Aware Prefetching

The idea is to transfer information, which is probably needed by the user in the future, so that it is already stored on the user's mobile device when it is actually accessed. This brings two major advantages. First, the system has lower response times because more data is available from the cache. Second, there is less "burst" load placed on the network because prefetching is done only when there is sufficient bandwidth available, rather than on demand.

The prefetching or put in reserve implies the prediction of the information which the user will request in the future. Instead of going to seek this information in a remote server, it will find them in the local cache of its mobile device. However, the approaches differ according to the objective of this operation of Tait, Lei, Acharya, and Chang, (1995). Indeed, the prefetching is used to improve the performance for the data storage used in the case of the disconnection. Moreover, it makes it possible to save the pile or the communication cost and when a user is disconnected because of the network low coverage, the user can choose the disconnected mode, Lee, Xu, and Zheng, (2002).

Before the prefetching one uses techniques which required the intervention of the user. These techniques did not have any success in the experiments because they make the user less sociable, and because generally the user does not have any idea of the subject of the information which he will need in the future of Zerioh, El Beqqali, and Laurini (2004) and Kuenning, and Popek, (1997) and (2002). In the other hand, the automated retention or the prefetching is the process to predict the information with the user will need in the future of El Garouani , El Beqqali, (2006) and (2007).

Data provided to information system users is defined by a multi-dimensional parameter space. Two dimensions define the geographic area of interest to the user and its current location, while the others represent the content describing attributes. Prefetching is in a way filtering all the potential information within the multidimensional information space against the parameters and then downloading the remaining data. Knowledge about a user's habits, preferences and interests is indispensable to compute the content important to a user. The mobile user needs to explicitly influence the query process whenever possible. This includes choices about the movement patterns, as well as the interests and preferences.

The prefetching aims at reducing the latency to get refreshed information appropriated to the current location. Thus, in order to effectively limit the prefetched information into the most likely future location context, the rectangle-shaped Prefetching Zone (PZ) is firstly defined with a user's moving speed and direction of Park, Kim, and Cho, (2004).

Kubach and Rothermel. (2001) the prefetching has been largely used for reducing latency in mobile information systems. In location-dependent systems, a change in the user's location requires refreshing its pending queries responses before sending the answers. This increases the server overhead and consequently increases query latency. This makes prefetching indispensable in location-dependent systems and mobile environments of Krummenacher, (2004).

Prefetching data for information management demands garbage collection. The best prefetching strategy has no use without intelligent data caching on mobile devices. Due to the limited storage space available on most of today's mobile devices, there is only a relatively small cache memory offered. Thus, a location and data-aware cache invalidation scheme is defined to support the prefetching mechanism. In addition to the invalidation scheme an efficient storage structure and content investigation method must be provided. Only with a cache implementation well suited for the mobile environment and collaborating with the prefetching scheme the impression of wireless links with high bandwidth can be created.

For location-dependent systems, prefetching is more essential for improving performance, because it is the only way to avoid the need of refreshing previous answers in reaction to location changes and to reduce query latency due to wireless systems limitations.

The problem of latency is crucial for local queries of Zheng, Xu, and Lee, (2002) whereas for non-local queries user movement for a short time does not invalidate the response. Must prefetch the information related to the current position of the user. This is the first requirement that our mechanism must satisfy. We formulate the requirements of a good prefetching technique for location-dependent systems and mobile environments as follows:

- Information related to the user's current area must be prefetched first,
- Closely related data must be grouped together,
- The user's direction must be taken into account; if the user moves to the North, prefetching data related to areas behind him (South) have no sense because he will never need this information.
- A cache invalidation scheme must be provided for freeing space from non-relevant data and prefetch more relevant ones,
- Prefetch only relevant information for not wasting bandwidth and the mobile device resources,
- The operations involved in the prefetching technique must be adapted for mobile devices of limited resources.

2.3. Cache Invalidation Strategies

The cache invalidation strategies in location-aware systems are certainly depending on the locality context. They are moreover very much linked to the prefetching strategies.

Use in their work about active query caching LFU (Least Frequently Used)/ LRU (Least Recently Used) approaches of Luo, Naughton, Krishnamurthy, Cao and Li (2000). They consider the access frequency and use of stored queries. Working in that way in the context of mobile information systems is rather complicated. It is difficult to determine the relevance of queries with respect to the changing user vicinity. It makes rather sense to take into account the users surroundings for decision-making.

The data catalog aims at using the advantages of both approaches to optimize the data management. First, the data descriptions contain semantically related entries, location or information type records that are easily compared. In other words the relevant information is filtered out of the queries ones to be used for the whole lifetime of a cache entry and stored as a block in the content directory. In that way cache invalidation and containment checking becomes straightforward, as the mechanism simply compares the location related attributes of the concerned data. Secondly, the data catalog does not do any semantic region management, as the content describes blocks of data belonging to the same query and not to the same semantic area. In consequence overlapping might still occur. A further advantage of a data catalog is the possibility to dynamically append additional information to the blocks to better describe the misused content.

By now the structure and the use of the content directory were discussed. An important part of cache processing still not discussed is the invalidation of disused data. It is necessary to remove old data to be able to store newly fetched and updated information. As the relevance of data is very much linked to a user's movement pattern the location is a principal clue for cache invalidation. Moreover users are generally conducting one-directional trips, rather than traveling constantly up and down a river. Therefore it is assumed that locations that were passed are no longer of considerable interest.

3. Description of Suggested Technique

In this step, we suppose that the geographic area covered by the system is subdivided into a number of adjacent squares. These squares set up groups. Each one has an access probability, and contains a precise number of items (files, tables, fields in a table, Web pages, video file,…). The data in each square are divided between cold and hot data (the first 20% of items are regarded as hot data and the

80% remainders like cold data) whose respective probabilities of consultation are 90% and 10% of Cao, (2000) and (2002).

Each square of the used area has length 1.

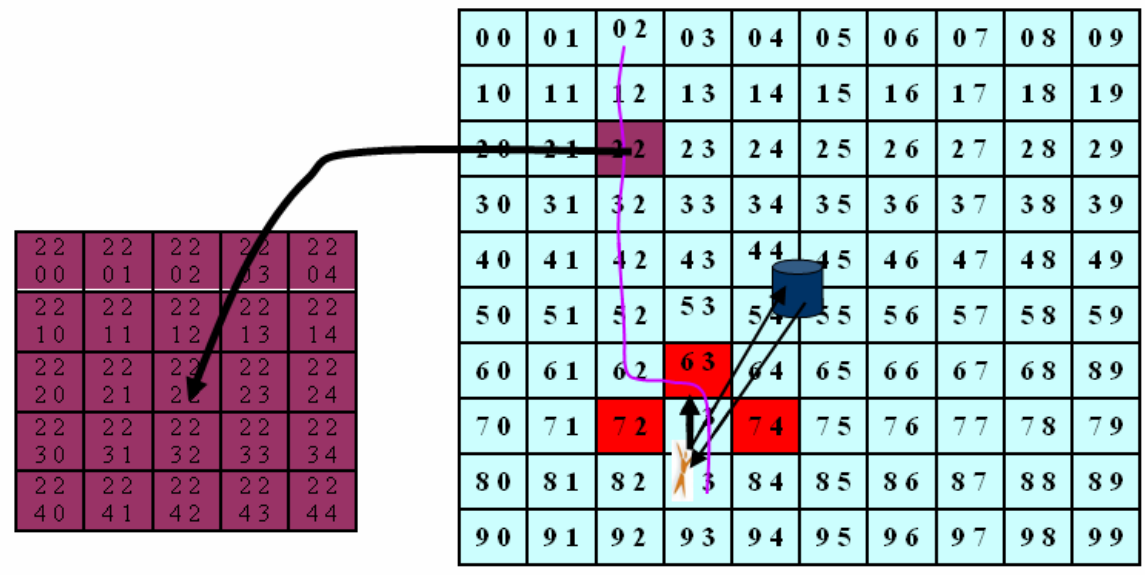
The resolution of the mosaic used in our simulation is N*N with N as integer.

For the direction of the users, we suppose that each user can take the following directions: North, South, East, West. And each user is characterized by position P, direction D, and speed V. We use a function of these tree parameters for the cache invalidation.

3.1. Probabilities Related to the Geographic Area

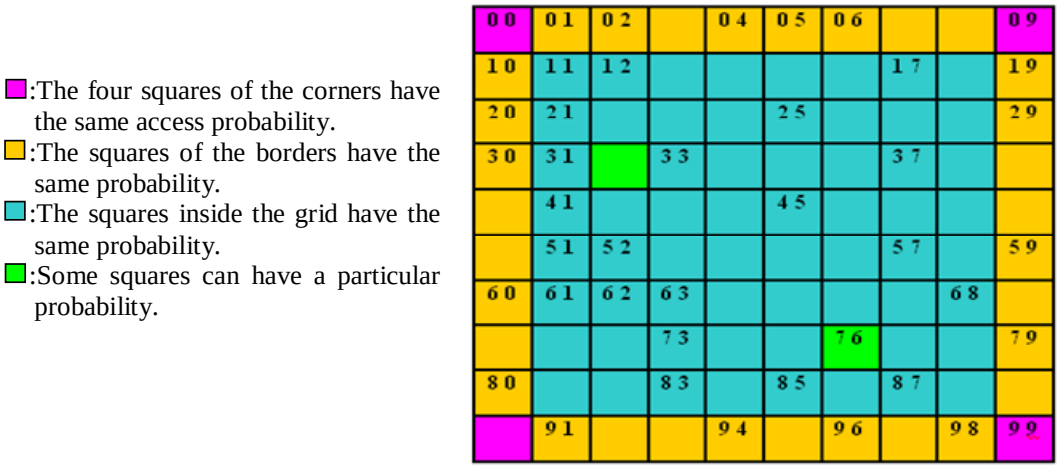
Mobile devices such as PDA are characterized by low storage capacity. So it is impossible to send client information about a large area. Therefore, the region will be divided into several squares and square identical in size, figure 1 presents an area squared divided into squares.

Figure 1: Area Divided into Squares and in Sub-Squares



For the proposed mosaic we use an access probability related to each square to decrease the quantity of data in the user's cache device. The possible probabilities are presented in Figure 2.

Figure 2: The Mosaic Possible Probabilities



3.2. Probabilities Related to the Users

For the users, each one has his own profile. Information presented by the system takes into account the preferences and the user's interest. For that, the system has the profile of each user describing his interests, his capacities and characteristics (position, direction, movement speed) and the used means of displacement (walking, bicycle, motor bike, car). Moreover, each user is pertaining to a user group or not. Each group is defined with only one profile. This user group built from the other like-minded users (the user with the same interests) could be helpful to improve the Information Retrieval (IR) systems effectiveness. The idea of Collaborative Information Retrieval (CIR) is to store the search processes of like-minded users in an archive. Subsequent users with the similar interests and queries should benefit from the knowledge automatically acquired by the CIR system, based on the stored search processes. A function of checking of the profile makes it possible to have a stereotype where the user is classified.

3.3. Algorithms

In this section we present the algorithms developed in the server and in the client, Figure 3 shows the architecture of the proposed prototype, PrefetchingManager determines whether the prefetched data ore necessary following the user profile, the search data to send preretching, transferring data in a way that the hot square adjacent to the position of the client will be sent first, and will keep track of data prefetched in order not to send further response to a question. The two paragraphs below show the program implemented at both server and client sides.

Figure 3: Model of the Proposed Prototype

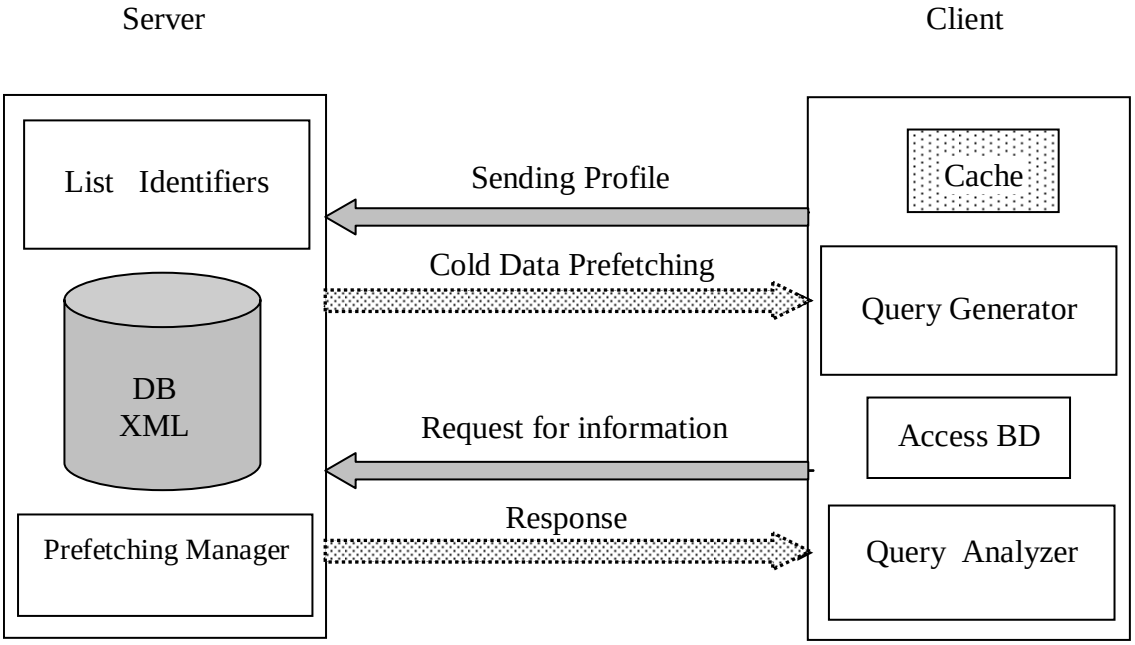


Figure 4: Sequence Diagram of Information's Request

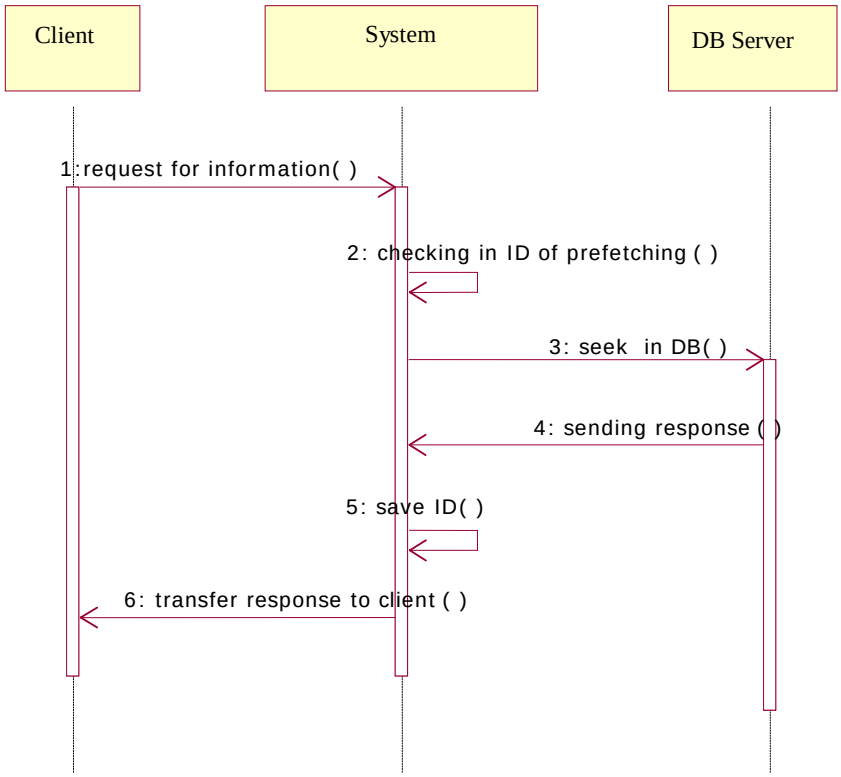


Figure 4 shows a scenario of the request for information: the client send request for information, the server checking in ID of prefetching and seek in DB, the DB server sending response with server who save Id and transfer response to client.

3.3.1. Algorithm in the Server Side

- The server receives the messages and determines if the message indicates a notification of place change (a request of the client which informs the server of its profile) or a message which requires specific data by specifying the profile.
- If the message is a question, the server checks the answer sent with the prefetched data, if not the response from the data base will send to the client.
- If the message is a notification of place change, the server sends to the following client its direction, the data of the three squares where the user will be in each one of them in his next displacement, and keeps list of the identifiers of the send item in its cache.
- The cache invalidation and prefetching criteria are executed.
- Elements related to ejected squares are dropped from the list of prefetched elements.
- The list of items to prefetch is sorted in a way that data related to the adjacent squares to the current square are sent first.
- The server begins sending the elements of the list of items to prefetch.

This algorithm describes how the directed prefetching strategy is implemented in the server side. The server is continuously listening for incoming messages (the new position to which the user moved and queries received from the client). When a query is received, the server stops the prefetching thread for sending the answer immediately to the client. A list of already prefetched information is maintained by the PrefetchingManager. Before answering a query, this list is checked. If the prefetched data do not contain the answer, this latter is sent to the client. When the user enters in a new square, the first thing that must be done is to stop the Prefetching Thread if this latter is sending data. As the prefetching criterion must be applied, there is no need for continuing sending data until the new prefetching set is determined. If this square belongs to a new square, the cache invalidation criterion is

also executed. Then, squares invalidated by the cache invalidation criterion are discarded from the list of prefetched elements. Intuitively, there is no need to execute the cache invalidation criterion in the server, because the client is responsible of its cache replacement policy. We choose to execute it also in the server for maintaining the list of prefetched elements. This latter reflects the client's cache content and is useful for deciding whether the client got an answer for the currently processed query from the prefetched data or not. Another important advantage of this list is not to resend already prefetched information related to squares not totally prefetched due to a position change that occurred while the prefetching process was still active. In the case where the user enters in a new square when the server is still prefetching data, based on the list of prefetched elements, the already sent elements are discarded from the set of elements to prefetch. The new set is arranged in a way that the elements related to the squares that are adjacent to the user's square are sent is not the non-adjacent ones. At this stage the server can begin sending prefetched data.

3.3.2. Algorithm in the Client Side

- When a query occurs, the client looks for the answer in its local cache. If the cache does not contain the answer, the query is sent to the server.
- When the client enters in a new square, the cache invalidation criterion is executed.
- When a message is received from the server, the client determines whether it is a response to a query or it is prefetched data. In the latter case, the client checks if it answers one of its pending queries before caching it.

The algorithm given above describes the steps executed by the client within the directed prefetching.

The client informs the server of its profile. The invalidation criteria of the cache were carried out to drop the data of the squares where the user will not be inside according to its direction to increase the memory for the relevant data. The user site and the direction are the most important criteria for the cache invalidation of its device Tian, Cai and Chin, (2001).

As a produced question, the client seeks in its cache, wether there is the answer, if not, the request will be sent to the server.

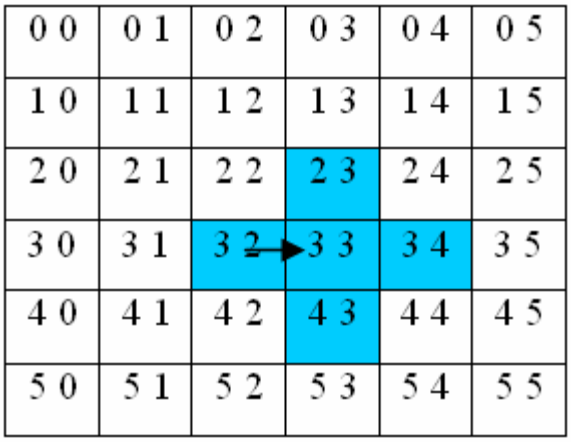
When a message is received from server, the client determines if it is a response to one of these questions or it is the prefetching data.

The client executes the cache invalidation criterion when the user moves to a new square. The task of sending the position may be removed from the algorithm, because as noted before, in location-dependent systems this is always done and is not only inherent to the proposed mechanism. For answering queries, the client looks for the answer in the cache before sending it to the server if the response does not reside in its cache. When a message is received from the server, it is first determined if either the message is a response or a previously sent query or a prefetched data from the server. In the latter case, the client checks if the data can answer one of its pending queries before caching this data. In this way, the client will not have to wait longer time for the answer from the server if this prefetched element answers a pending query. At the same time the server will not have to re-send a previously sent element as noted before.

3.3.3. Access Model to the Information of the Suggested Technique

We estimate that the server is able to manage a great quantity of data of the considered area. Before each displacement, the user informs the server of its profile. The server sends relevant data to the client of the squares where he could be in its next displacement (Figure 4).

Figure 5: Data in the cache of the client following his direction



4. Experiments and Simulations

4.1. Prototype

At first, the development of an approach consists in storing information (files, tables, fields in a table, Web pages, video file, ...) in XML database. In the tourism's field information, are automatically generated of tourist surveys, and survey data are collected in XML documents to be stored to be interviewed later. We develop a Java application based on the use of threads to access the data in this XML database.

We describe our simulation model with Java (SDK1.4.1) and a prefetching prototype implemented to show the performance of our approach compared to prefetching area and prefetching space at the mass of data stored for each user and for the area where it is a slow response. In the prototype, we implemented an application that allows tourism potential clients visiting tourist sites to receive the data they need in the future movement, by the server according to their positions, speed and direction. In particular it will make location requests such as: "What is the nearest five-star hotel? ". In this prototype, the various potential clients are equipped with mobile devices with a wireless card 802.11b/g. The server that is available is also equipped with a WiFi card. The shared data is stored on the server in the form of XML files with their geographical coordinates. The data server may be files, tables, fields in a table, Web pages or video files,

4.2. Analysis

To evaluate the proposed solution, we have tested in a WiFi network using laptops and PDA with the following situations:

- Connected Environment: several users exist, and each participant may not be able to communicate with other participants.
- All locations are expressed in physical data: This location can be estimated using our algorithm, or given by a geo-technology location (GPS for example).

To determine the best way, our technique was evaluated for multiple users to compute the average costs in terms of response time associated with evaluating the request, communication and energy. To test these costs, we have developed distributed between server and client.

In fact, test the technique using the prototype we have developed over the mass of data stored for each user and for the state along the answer.

These simulations are performed to assess the various costs involved in the evaluation of the request dependent on location. Unlike the optimization in databases, we do not consider optimizing the local assessment of the application on a node but more generally the costs associated with distribution of the application in the implementation of prefetching.

On evaluation of a local query on a mobile terminal, various works such as Anciaux, Bouganim, and Pucheral .(2003) propose solutions designed to minimize the use of resources during a query on a terminal forced.

4.3. Description of Different Application's Components

In our application, the client program is installed on each client device; a version of the prototype is deployed on each device. The architecture of the prototype implementation is described in figure 6, and thus the prototype based on four main components:

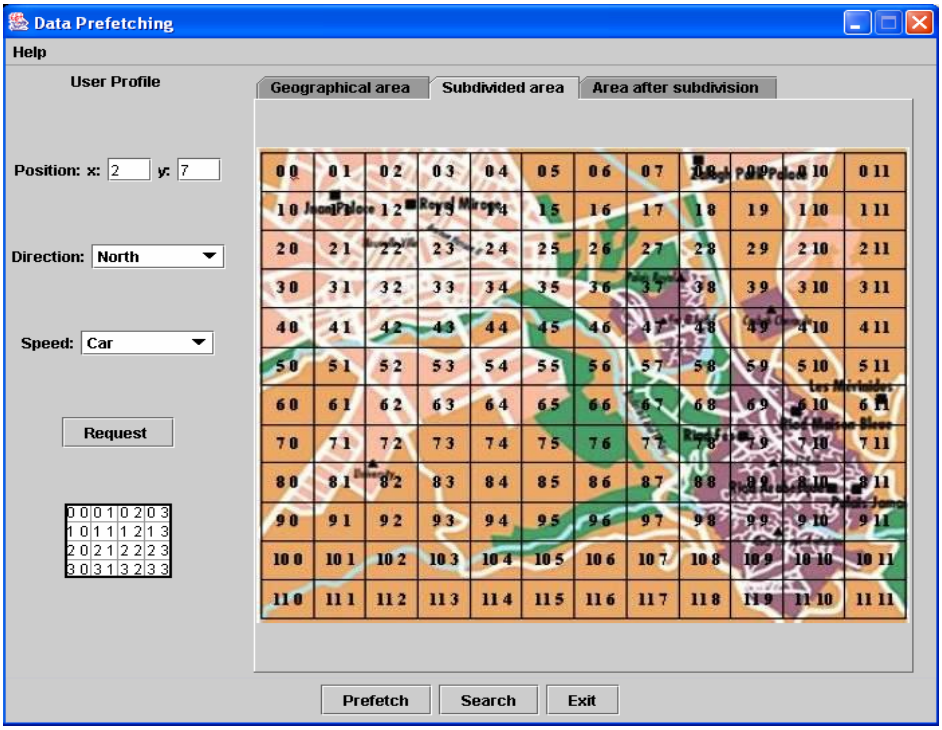
- User Interface: provides the user with a means to compose his complaint and see different results.
- Storage Manager Information shared by the user.
- The positioning module: provides client location's.
- Assessor queries prefetching: the location allows for the dissemination of information by clients but provides an evaluator dedicated to the evaluation of requests made by various participants in implementing prefetching.

When a client makes a request, the client starts typing a request via the user interface found on his terminal. The request before it is sent to the server. If a request is dependent on the location, the server interrogates the positioning data table to retrieve the geographic location of the user. Then, the evaluation of the complaint is managed by the service that takes into account the client's direction, and speed to load data from neighboring square (prefetching).

4.4. User Interface

In the prototype, two sets of different interfaces have been made. The first is aimed at users equipped with portable PCs and the second is adapted to light devices (PDA, ...). Indeed, for a tourist, features offered to different nodes are different depending on the type of nodes (central or light). Similarly, constraints on display are different and therefore involved in the design of different interfaces, figure 6 shows the client interface for a grid of size 12 * 12 squares.

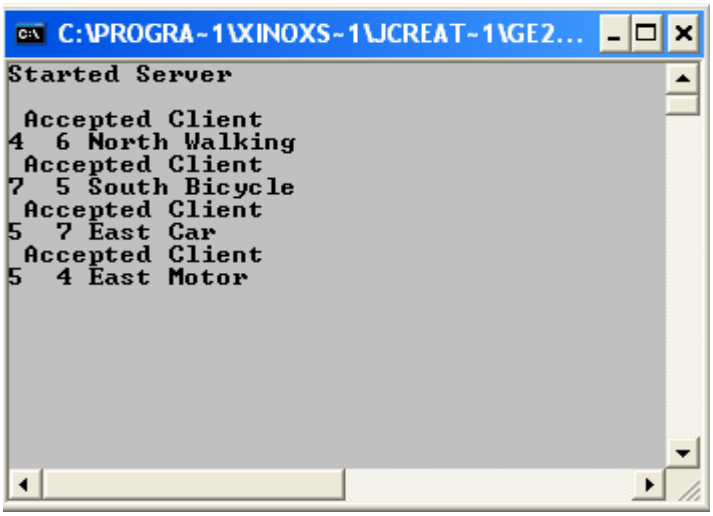
Figure 6: Interface application for the client



4.5. Server Interface

In the developed prototype, the server-side interface allows users to be connected to the server with their position, direction and speed. Figure 7 shows the interface on the server side.

Figure 7: Interface for the application server

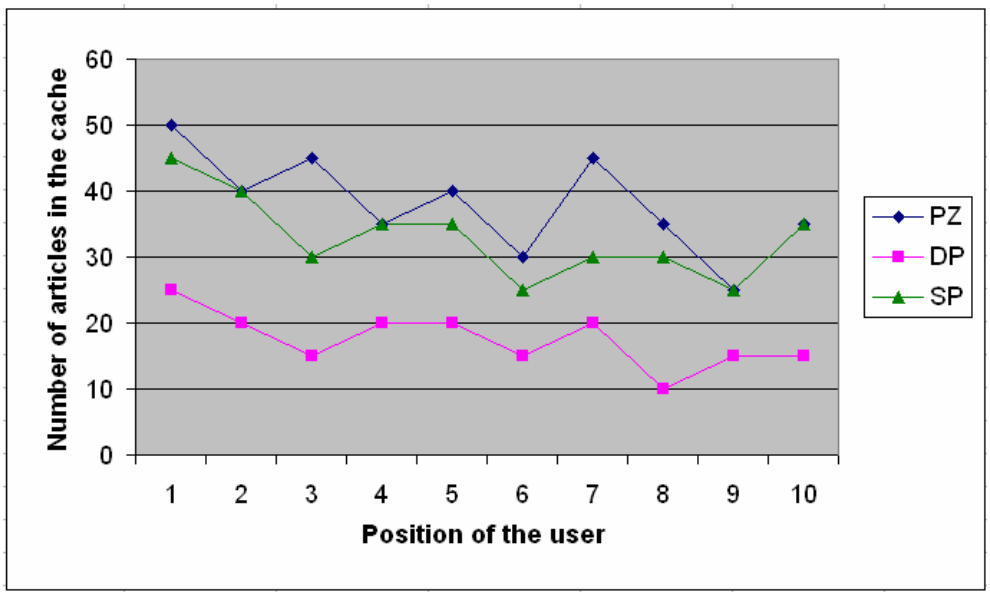


4.6. Comparison of Our Technical Report to Spatial Prefetching (SP) and Prefetching Zone (PZ).

The comparison represented in figure 8 shows the contribution of the proposed method we called "Directed Prefetching." (DP) Indeed, compared to the other two methods Spatial Prefetching (SP) and Prefetching Zone (PZ), it gives better results about the rate of success, compared to the volume of data stored for each user: it decreases the mass of data, the squares in bold show data cache mobile device user in each technique.

The curves figure 8 shows the difference between the three techniques in the evolution of the number of items pre-loaded into the cache of mobile device a client made a journey from the slot 1 up at the location 10 in a grid of 12*12 square with a number of articles in each square 20 (5 hot colder and 15), and the cache size of its 50 articles at most.

Figure 8: The number of items in the cache of clients over its location.



5. Conclusion

We have presented a prefetching technique that deals with the complexity of real systems. No assumptions were made about the future user's movement. Inspired from previous work in general mobile systems, we derived the basic requirements for a successful prefetching scheme for location-dependent systems. Our mechanism prefetches data related to the user's location as it is the most crucial for local queries. This is realized by dividing the squares into sub-squares and we take the user's direction into account and avoid prefetching for non-relevant data. The space is also divided in squares and a first implicit grouping of related data is done. The grouping is even smoother by using the hypermap structure of Zerioh, El Beqqali, and Laurini (2004). Associating the average probability of the access to each document node, avoids sending information that will not be requested and allows to prefetch a fixed amount of information to satisfy the available bandwidth and available memory in the user's device. The cache invalidation criterion proposed, ensures that non relevant data do not waste resources unnecessarily. An other advantage of this scheme is that the user is always provided by up to date information because it is continuously refreshed by executing the prefetching and the cache invalidation criteria. An important benefit of this is to avoid extra communication for checking cache consistency. We have proposed an implementation algorithm that delegates the hard tasks to the server for making the method implement able in the mobile devices with the poorest resources. Thus, a prefetching technique which takes account of the user site to send him the relevant data and to reject those not relevant was developed. It also makes it possible to take into account:

First, the treatment in the context where several users reach the same information and at the same time, and second, the profile of each user (position, direction, speed, etc). This last is an important parameter for this technique, because it takes account of the requests number at the same time, and we presented the protocols and the scenarios of our technique.

We project following this work to propose a more general prefetching architecture. This architecture must take into account all the hierarchical proxies caches or collaborative distributed in the area covered by the system, and allows also increased other parameters compared to the other prefetching techniques. We also consider algorithms side client and side server and/or proxy allowing taking into account the probability related to space and the users with their profiles.

References

[1] Anciaux N., Bouganim L., Pucheral P., 2003 “Memory Requirements for Query Execution in Highly Constrained Devices”, 29th Int.Conf. on Very Large Data Bases,.

[2] Badrinath, B.R., Imielinsky, T., Frankiel, R. and Goodman, D., 1996. Nimble: Many-time, many-where communication support for information systems in highly mobile and wireless environments, http://www.cs.rutgers.edu/~badri/dataman/nimble/.

[3] Bernard G., J. Ben-Othman, L. Bouganim, G. Canals, B. Defude, J. Ferrié, S. Gançarski, R. Guerraoui, P. Molli, P. Pucheral, C. Roncancio, P. Serrano-Alvarado, P. Valdurie. 2001 Mobilité et bases de données, *TSI, Volume X°* ,.

[4] Cao, G. 2000 “A scalable Low-Latency Cache Invalidation Strategy for Mobile Environments”, Proceeding of ACM MobiCom, Aug., pp. 200-209.

[5] Cao, G., 2002 “Proactive power-aware cache management for mobile computing systems”. IEEE Transactions on computers, , pp. 608-621.

[6] De Nitto, V.P., Grassi, V., Morlupi, A., 1998 “Modeling and evaluation of prefetching policies for context-aware information services”. In Proceedings of the 4th Annual International Conference on Mobile Computing and Networking, (Dallas, Texas, USA), pp., 55-64.

[7] Dik Lun Lee, Jianliang Xu, and Baihua Zheng, 2002 “Data Management in Location-Dependent Information Services”, IEEE PERVASIVE computing, , pp. 65-72.

[8] El Garouani S., El Beqqali O. 2006 “Optimisation de stockage et accès aux données dans le prefetching dans un environnement mobile”, IA Oujda, Maroc, 2006, pp 153-157

[9] El Garouani S., El Beqqali O. 2007 “le critère d'invalidation du cache dans le Prefetching" 7ème Conférence Internationale NOTERE'2007, 4-8 Juin 2007. Marrakech, Maroc,

[10] El Garouani S, El Beqqali O., and Laurini R. 2007 "Cache invalidation method for prefetching in mobile environment", on Information and Communication Technologies International Symposium (ICTIS’07) IEEE, Fez Morocco,

[11] Kian-Lee Tian, Jun Cai and Beng Chin Ooi, 2001 “An Evaluation of Cache Invalidation Strategies in Wireless Environments,” IEEE Transactions on Parallel and Distributed Systems, vol. 12, no. 8, August, pp. 789–807.

[12] Kuenning, G. H., and Popek, G. J., 1997 “Automated hoarding for mobile computers. In Proceedings of the 16th ACM Symposium on Operating Systems Principles”, (St. Malo, France),

[13] Kubach, U. & K. Rothermel . 2001: Exploiting Location Information for Infostation-Based Hoarding. Proc. Seventh Ann. Int’l Conf. Mobile Computing and Networking (MobiCom’01), Rome, Italy, pp 15-27.

[14] Kuenning,G. H., Ma, W., Reiher, P., Popek, G. J., 2002. “Simplifying Automated Hoarding Methods”, In Poceedings of ACM/MSWiM’02, (Atlanta Georgia USA)

[15] Luo, Q., J. F. Naughton, R. Krishnamurthy, P. Cao & Y. Li 2000, “Active Query Caching for Database Web Servers”. Lecture Notes in Computer Science Vol. 1997/2001 (Third Int’l Workshop WebDB Dallas, TX, USA), Springer Verlag: pp. 92-104.

[16] Reto Krummenacher, 2004 A Location-Aware Prefetching Mechanism, Swiss Federal Institute of Technology CH-1015 Lausanne,

[17] Seungmin Park, Daeyoung Kim, and Gihwan Cho, 2004 “improving prediction level of prefetching for location-aware mobile information service”, Future generation Computer Systems, Volume 20, Issue 2, 16 February, pp. 197-203.

[18] Tait, C., H.Lei, , S. Acharya, Chang, H., 1995 “Intelligent File Hoarding for Mobile Computers”, In Proceedings of the first intrnational Conference on Mobile omputing and Networking, ACM Press, (Berkeley USA),.

[19] Zerioh K., El Beqqali O., and Laurini R., 2004 "A Hoarding Strategy for Location-Dependent Systems" In Fisher Peter F. (Ed.) Developments in Spatial Data Handling. Springer-Verlag, pp 217-230

[20] Zheng, B., Xu, J., and Lee, D.L., 2002; “Cache invalidation and replacement strategies for location-dependent data in mobile environments”. IEEE Transactions on Computers 51, 10, pp. 1141-1153.

[21] Zhou, H., Feng, Y., Li, J., 2003 “Probability graph based data hoarding for mobile environments”, Information and Software Technology, 45, Elsevier Science, pp. 35-41.