A Data Allocation Scheme using Data Mining for Wireless Cellular Network

John Tsiligaridis  
Mathematics and Computer Science,  
Heritage University, 3240 Fort Road, USA  
Toppenish, WA, 98948  
(tsiligaridis_j@heritage.edu)

Raj Acharya  
Computer Science and Engineering,  
Penn. State University, USA  
220 Pond Lab., University Park, PA 16802  
(acharya@cse.psu.edu)

ABSTRACT

A centralized collaborative system between nodes, servers and Base Stations (BSs) is developed, and a new data allocation scheme based on group and query prediction mobility scheme with Data Mining techniques is proposed. With this approach we minimize the cost of remote access by reducing the number of network messages and improve the response time of the queries. A series of algorithms is developed so that the system can provide the “prefetch” operation for the group of users according to their mobility. The data have to follow the users’ moves. Our new approach contains Call Admission Control (CAC) with a prediction scheme for Group and Query Mobility (GQM) based on two phases: the Merge Itemsets Algorithm (MIA) and the one step dynamic Replication Algorithm (RAL1) respectively. The RAL1, providing two data replication policies, can guarantee the acceleration of the query execution. The proposed system can be used for site selection or reduction of the number of sites in query optimization, and also the design of any new architecture for distributed query processing. Simulation results are provided.

KEYWORDS: Mobile and wireless computing, Data Mining for Collaborative Technologies, Mobile Database, Distributed Databases, association rules

1. INTRODUCTION

An increasing number of users access various types of information via their mobile devices. Mobile users are able to receive and send e-mail from and to any location as well as be informed about certain predefined conditions (information services). Stock information will be provided according to user portfolio and weather report, traffic conditions are provided based on the current position of the user. We will explore the approach of mining user moving patterns in a mobile computing environment and utilize the mining results to develop a data allocation scheme. The use of moving patterns in the data allocation scheme can diminish the occurrences of remote access. The frequently visited servers must hold the important information available at the time they are asked for from a group of users. The mobile computing system has usually the distributed server architecture where the servers can provide services to mobile users of their area[1]. The whole geographic area is partitioned into cells. Each cell is covered by a base station attached to the fixed network and provides communication link between the mobile users and the nodes of the network. Users usually submit transactions to the servers of their area and receive the answers in the same or another nearby server that the user probably will move towards. Two types of data objects are accessed by the users: the “read” and the “update” ones. In addition, an homogenous distributed database system is considered where all the sites with identical database management system software are aware of one another and agree to cooperate in processing users’ requests[2]. Data objects are stored in servers in order to save memory space of the mobile device and also facilitate the maintenance of consistency of data[3]. A consequence of the distribution of data is that a transaction may involve processes at several sites. The most important advantage of the mobile computing system is the ability to replicate data, that is, to make copies of data at different sites. The motivation of such a process lies on the fact that if one site fails the other sites can provide the same data. A second use is that there is an improvement of the speed of query answering by making a copy of needed data available at the sites where queries are initiated[2],[4],[5]. The approach of data replication increases the overhead on the data update and storage[2],[1]. The system must ensure that all replicas of a relation are consistent otherwise may take erroneous computations. The update operation must be propagated to all the other sites. In this work we do not deal with the case of data fragmentation (horizontal or vertical). We focus on the group mobility and the correspondent query execution path (qep) in order to minimize the communication cost, and the response time.
Many data allocation schemes have been developed [4],[1]. In [4] the data allocation scheme was designed for traditional distributed databases in static manners with service areas not fully explored. In [1] data allocation schemes have been presented in various scenarios using the incremental Data Mining technique, but without consideration of the time domain. Thus there is no accurate design for the prediction of the time when a user will most probably be found in the subsequent cell in order to receive the updated data. In addition in [1] is considered that all the data have been gathered only in one site; a condition not existing in the distributed databases. A user who has relocated and has been using certain files and services at his previous position wants to still have his environment or the results of the requested services recreated at the new position [6]. Of course this is particularly important for the maintenance of performance transparency which can guarantee similar access and latency time independently of the user position. In this work we notice that the users move along the cells with patterns of read/write activity and the data are considered to be spread in various servers. The mobile environment needs dynamic replication schemes that will replicate data according to user or group mobility or in other words on the basis of changing “centers of activity”. The data should follow the users. For our purpose we first present an algorithm to capture the frequent user moving patterns. This is achieved by Data Mining methods discovering first the user or group mobility patterns and then the data allocation scheme providing a dynamic replication algorithm (RAL1) for optimization of the distributed queries. The CAC defines the time period and the site in which the system has to make the data replication.

The paper is organized as follows. In chapter 2 the model description is developed. Query and mobility are developed in chapter 3. The Description of the Group and Query Mobility (GQM) with The Merge Itemsets Algorithm are described at chapters 4,5 respectively. The RAL1 with the CAC are presented at Chapters 6,7 respectively. Simulation results are provided in chapter 8.

2. MODEL DESCRIPTION

2.1. Wireless Network Model

The configuration of our system consists of a fixed information network extended with wireless network elements. The classical wireless/mobile network with a cellular infrastructure with the standards of GSM and IS-41 is considered. Our approach can also be used for Unified Mobile Telecommunications Service (UMTS). A user resides in a cell and communicates with one Base Station (BS). BS is responsible for forwarding data between mobile users and wired network. Each user has a home database termed the Home Location Register (HLR) which is a part of the user profile. When a user moves to another location can also be registered as a visitor (VLR).

2.2. Data Mining concepts

2.2.1. The Basics - Rules

A Data Mining method is developed in order to predict mobility patterns by utilizing their previous history. For our group mobility we use the 2-itemsets. Let I = {i_1, i_2, i_3,..,i_m} be a set of literals called items. An itemset containing k items is called a k-itemset. A transaction T (or itemset) is a set of items such that T ⊆ I. Let X be a set of items, a transaction T is said to contain X if and only if X ⊆ T [7]. An association rule implies the form X => Y, where X ⊆ I, Y ⊆ I, and X ∩ Y = ∅.

For the association rules there are two basic measures, the support and the confidence. Support of an itemset (string) stands for the number of times that an itemset or sub-itemset appears in the database. The task of mining association rules is essential to discover strong associations rules in large databases. The prediction results are used to reserve bandwidth in advance of a connection with eventually, adjacent cells. Each item represents a cell. The path is as an itemset or a large itemset (l-itemset), that is a set of items (numbers or letters). Common subpath (cs) is the common part of the branch paths. The cs of “abcde” and “abcdef” is the subpath “abcd”. Restricted Support is the support given by the first same items that compose the cs. Extensive support is given by extension of the support operation beyond the first cs. The confidence c of the rule is computed as the quotient of the supports for the itemsets c = sup(A,B,C,D) / sup(A,B,C). Strong association rules are rules with a confidence value c above a given threshold [7].

2.3. Distributed Databases

2.3.1. The Basics

Many schemes have been devised for interpreting relational query languages in distributed systems. These schemes are essential strategies for executing a combination of projection, restriction, and join operators from relational algebra in order to answer a query. Two of these methods, the distributed retrieval algorithm for SDD-1 and the RAP database machine use a special composed operator (semi-join) for join and projection.

2.3.2. Query Processing

The principal problem for the evaluation of a query on distributed databases is when two relations must be joined and while they reside in various sites. For the join
operation one relation from the one site must be shipped to the other site. Since the communication is the principal cost for the distributed databases the minimizing of the amount of shipped data is of first interest. Of course it is not possible to eliminate the communication costs completely but it is possible to minimize it significantly. There are two main methods for querying processing. The first is when we have local processing with all the restrictions applied and the second is when we use join or semi-join [2].

Example 1: For r1 (relation) on site s1 and r2 (relation) on site s2, the projection of r1 on its joining column on the r2 using semi-join to reduce r2 by r1 before shipping to the s1. A sequence of the moves can be: from s1 to s2 and finally the results are taken back to s1. Sometimes it is better to read data from the servers’ disks (as in a hybrid-shipping). The best way to execute queries that have shared data at two different servers might be to read both tables from the servers’ disks and to execute the join at the client. For the query optimization plan the site(s) selection is the most important factor because it provides the information of where the query should be submitted, when and which part of the data should be optimized [8].

3. QUERIES AND MOBILITY

The easy case occurs when the answer to the query can be given locally. The data (for one person or group) can be prefetched at the next site according to the prediction of mobility. The advantage of the mobility prediction is fully exploited since the response time is reduced from the later access.

Example 2: A user or a group (with “ABC” path) is moving from A to the next location B and the data according to the path follow him at B. During the execution of the query in various sites, the mobile can stay or move into another location. Generally, when the user remains at the same location (static environment) the parts of the distributed queries, until the integration of the execution can create a cyclic graph. The query starts the execution from a cell (A) and after the shipping of the data (or the intermediate results) to the other sites, the final results are sent back to the same cell (A). The study of the various ways of execution of the distributed queries (including the cache mechanisms) is beyond the scope of this work.

Definition 1: The cell from where the user asks for execution of a query is named starting query cell (sqc).

Definition 2: A query execution path (qep) is the sequence of servers at various sites that this query have to follow using the join or semi-join operation. The size of qep (s_qep) is determined by the number of the servers of a qep.

Definition 3: The last server where the query results are finally ready and the previous of the last server of a qep are called ls and pls respectively (f.e. for the qep “BEG”, ls = “G”, pls = “E”).

Definition 4: A server (s1) is predecessor of any other server (s2) if the query first visits the s1 and then the s2 or if ord_s1 < ord_s2

Example 3: A user is moving according to the path “ABCD”. At cell B he sends a query which can not be answered locally (from the local server) and the data are sending to other cells not included in the user path. The path that the data follow is: “BEG”. After the integration of the query execution the results are sent back to the cell B, from where the updated data are sent to the new user location (cell D) according to the user mobility.

Thus, in this case the data can not be prefetched by the updated user location. To minimize the data delay the ls has to send the data directly to the new user location. This is provided by the CAC. It would be very helpful for the data, the query and for any hybrid shipping operation if the data were not spread out to many sites. An effort is made to diminish the number of sites for data tuples or relations by replicating the data from the one site to the other before the execution of the query.

The cost of a query execution consists of two phases: (1) the execution of all the parts of the queries according to qep, (2) the return phase (the data are sent to the sqc).

Here we focus on minimizing the qep by one step and sending the final data directly from the ls to the new update user (or group) location.

4. THE DESCRIPTION OF THE GROUP AND QUERY MOBILITY (GQM)

We use a collaborative system (cell-BS-server-node(or high level servers)) with the composition approach, in order to take the mobility trend (periodical events) without taking into account the user mobility profile. The GQM based on the groups’ move, can guarantee the acceleration of the query execution. The BS creates the preproces sing phase. It gathers the tuples according to their arrival time and distributes them into a predefined time period (dt). Additionally, it gathers information for the query execution path. (time, sites, etc from Fig 1) Periodically or at the end of the day it follows the phase of sorting and transferring the tuples (with support>ms[7]) to the nodes for processing. The server of each qep, also sends the time and the support of the requested data to the node (or upper level server). The ls (that holds the final result) works the same as the server. The node works in two phases. In the first phase gathering the information of the group and query mobility
(in tables), finds first the paths in each BS and then among adjacent BSs. In this way it creates the most popular group path. This information is sending to the BSs (details ch. 5). In the second phase elaborates with the query optimization performance (replication of data, compression path), prepares the new query path, informs the analogous servers and especially the is about the time and the location the group’s results should be send. According to the new query path the local servers with the BSs regulate the data replication on the analogous sites before the execution of the query (details ch. 6,7).

Example 4: In Fig. 1 we assume that each BS is serviced by a server (hierarchical model). A group trend from cell a (BS1) to cell k (BS2) : a → b → d → j → k.(when no query is sent). A query starts from cell b with the qep: b-server (S3) ->server (S4) (1),(2). Finally the results from S4 are sent back to b following the reverse move which take time, cost and obviously the data do not follow the group’s moves since the group goes from BS1 to BS2. The nodes will inform the server (S4) to send the query results at BS2 (new group location). The replication of data from S4 (supp.=20) to S3 (supp.=60) could be another query optimization measure (path compression). Thus transferring the updated data directly to BS2 (using (3)), in combination with the path compression (using (4)) will be the best timely solution.

5. THE MERGE ITEMSETS ALGORITHM

There are more than one ways to connect the subpaths of two or more routing tables of the BSs. In this work we develop the Merge Itemset Algorithm (MIA) which takes place at the nodes. Connectivity is the possibility to connect 2-itemsets and this happens when for two 2-itemsets, the 2-itemset, (item$_{i1}$, item$_{i2}$) and the 2-itemsetj (item$_{j1}$, item$_{j2}$) we have: item$_{j2}$ = item$_{i1}$ . We also define the internal connectivity when the user moves inside the BS, (all the items or cells belong to the same BS), and external connectivity when the item$_{j2}$ ∈ BSj. The steps for MIA are as follows:

For each BS:
1. Find the internal connectivity, (increasing time period) until external connections will be reached. Create a group for a series of internal itemsets followed by the external itemsets. From Fig. 1 we take the group of items: “abd”, where: “ab” is the internal connection and “bd” the external
2. Find the confidence for each 2-itemset, and sorts according to the confidence (only when we have increasing or decreasing trend of moves).
After we finish this process for each BS we continue with the pair process.

For a pair of BSs:
1. Find the external connections for any itemset of the 1st BS with any itemset of 2nd BS. From Fig 1 we can see that the external connectivity is d->j for BS2.

![Fig. 1 The structure of node finding the group trend with the query processing](image)

6. THE ONE STEP REPLICATION ALGORITHM (RAL1)

Definition 5: One cell (move) meeting delay or next cell meeting delay is the case when the results of a query find the user at the next cell (move) from the sqc. Two or three cells meeting delay is when the results of the query find the user at the cell after two or three moves from the sqc. In this work the three cell meeting delay is considered as upper limit delay for any distributed query.

Example 5: Let us be a user path “ABCD”. The user sends the query from cell B and the qep is “BEFG”. If he/she finally receives the answer at cell D we have two cell meeting delay.

Thus, for a group of users the better query time response happens in one cell meeting and the worst query time happens in three moves. Depending on the time, the frequency of data requests from a site and the execution order of the different parts of a query at various sites, a one step replication algorithm (RAL1) is proposed. This algorithm reduces the data route by one server move, the communication messages, and the response time of later access. It is examined with various parameters: the minimum response time (pred_resp) and the one, two or three cells (moves) meeting delay. Two policies are provided for the reorganization of the data in the sites: (1) the maximum support site expansion (MSSE), where the data from the server with the minimum data support are replicated to the server with the maximum data support,
if the maximum support is greater than a predefined threshold (pred_max_supp) (2) the last site compression (LSC), when all the servers of the qep have support less than the predefined threshold (pred_min_supp), the data are simply replicated from ls to pls. In this work we only consider one step replication or in other words compression of the qep by one step. The compression is a way of minimizing the s_qep and decreasing the delay.

Replication algorithm (RAL1) //for qep > 1
//finding the data of cells that must be shipped to the previous location
//reduce (compression) the qep
//pred_resp: is the minimum response time,
//response_time: the response time of the query
//pred_max_sup: predefined maximum support
//pred_min_sup: predefined minimum support
for any qep (S1,S2, ...,Ss, with sup> m_sup) {
    if ((response_time> pred_resp) ||
        (there are 1,2 or 3 moves meeting delay))
    {
        reorganize the data,
        find server k with the max_sup
        max_sup_Sk = max support (supS1, supS2,supS3....supSs)
        find server m with the min_sup
        min_sup_Sm = min support (supS1, supS2,supS3....supSs)
        // MSSE
        if (there are k and m servers) && (max_sup_Sk > pred_max_sup)
            (min_sup_Sm < pred_min_sup) )
            replicate data from Sm to Sk //compress path by one step
        //LSC
        if (for all the i servers (Si) of a qep the supSi < pred_min_sup)
            replicate data from cl to pcl //replication of the cl
    }
}}

7. THE CALL ADMISSION CONTROL (CAC)

Furthermore after the RAL1, the Direct Access Transfer (DAT) of the query results from the ls to the new candidate group location will improve the total time of the query execution. A predictive Call Admission Control (CAC) is needed for the synchronization of the execution of the new query plan (after the application of RAL1) with the transmission of the results to the new group location. The CAC takes place at the nodes (or upper level servers) and the processing results (the new query execution plans) are sent to the servers that participate in the execution of the new query plan.

Example 6: From the example 5 the ls (“G”) having the final query results will send them directly to the candidate node D without following the path B->C->D.
The performance of the proposed algorithms is examined for various user mobility patterns and some scenarios are presented below:

1. All kinds of users (scenario 1). Queries with data in 1, 2, or 3 servers are considered. The pred_res = 10 sec (tight response time condition). The upper positioned curve refers to the case before the compression (BC) or before the CAC of s_qep. The minimum value of pert is achieved by applying the combination of MSSE, and DDT. (Fig. 2). In Fig. 3 considering only s_qep = 1 (only one server move) with the same pred_res we take lower pert values (than in Fig. 2) with the LSE policy. In Fig. 4 considering also s_qep = 1, with looser response time condition (pred_res = 3 min) we take zero pert values.

2. Users with branches and restricted support (scenario 2). A new set of user mobility patterns with more branches and the same parameters (pred_res = 10 sec, data in 1, 2, 3 servers) is considered. The upper curve refers to the case before the compression (or CAC). The better results are taken when MSSE, and DAT are applied. (Fig. 5)

3. Users with branches and extensive support (scenario 3). The same parameters are used as in Fig. 5. We have a little greater values of pert in Fig 6 than in Fig. 5 because of the greater size of the path. By contrast with the restricted support paths, here it is more possible for the user to move at a far away location, which adds an additional time to the response query time.

9. CONCLUSION

A collaborative scheme among nodes, BSs, and servers is presented. A set of algorithms discovers the group mobility that is the base for an optimization of the distributed query process which is achieved through the use of the dynamic one step replication algorithm. Considering these concepts along with the shared data in various locations’ servers, the use of RAL1 sounds very promising especially when we have query shipping (for a middle tier site), data shipping (for the general query execution) or any economic model for resource allocation (using brokers etc).
REFERENCES


