Allocation of Data Items for Multi Channel Data Broadcasting in a Mobile Computing Environment

Agustinus Borgy Waluyo[1], Bala Srinivasan[1], and David Taniar[2]

[1] School of Computer Science and Software Engineering, Monash University, Australia
{Agustinus.Borgy.Waluyo,Bala.Srinivasan}@infotech.monash.edu.au
[2] School of Business Systems, Monash University, Australia
David.Taniar@infotech.monash.edu.au

Abstract. Utilising broadcast mechanism for query processing in a mobile computing environment is very much desirable due to its scalability. Therefore, it is necessary to maintain the advantage of this mechanism by minimising query access time over a large number of broadcast items. In this paper, we propose a broadcast ordering scheme at the server side, which aims to minimize query access time of mobile clients. We concern with data broadcast over multiple wireless channels. Generally, this scheme considers single and multiple data items request. We develop a simulation model to analyze the performance of the proposed scheme. The proposed scheme provides a substantially lower access time as compared to the conventional method.

1 Introduction

Recent breakthrough of wireless technology has created a new era of database information retrieval. People are now able to conduct their business anywhere and anytime using portable size wireless computers powered by batteries (e.g. PDAs). These portable computers communicate with a central stationary server via wireless channel, and so are known as mobile computing [1, 12, 4]. A subset of mobile computing that focus on query to central database server is referred as *mobile databases.* Mobile computing faces a number of restrictions particularly power, storage and bandwidth capacity. With power capacity, the life expectancy of a battery (e.g. nickel-cadmium, lithium ion) was estimated to increase time of effective use by only another 15% [5]. Thus, efficient use of energy is definitely one of the main issues.

Broadcast strategy refers to periodical broadcast of database items to mobile clients through one or more broadcast channels. Mobile clients filter and capture their desired data over the channel. With this mechanism, mobile client is able to retrieve information without wasting power to transmit a request to the server. Other characteristics of data broadcasting includes scalability as it supports a large number of queries, query performance is not affected by the number of users in a cell as well as the request rate, and effective to a high-degree of overlap in user's request. Figure 1(a) illustrates broadcast mechanism. This mechanism is also referred as push-operation [3, 13]. One of the ultimate challenges in broadcast strategy is to minimize the query access time of the client to obtain information from the channel. Access time is defined as the elapsed time from the time a request is initiated until all data items of interest are received.

L.T. Yang et al. (Eds.): EUC 2004, LNCS 3207, pp. 409–418, 2004.

In this paper, we present a broadcast ordering scheme for multi broadcast channels. The proposed scheme is designed to make the most requested data items stay close to each other in the channel. Consequently, the query access time of mobile client to retrieve the desired database items is minimized. This paper concerns with single and multiple data items request.

Furthermore, we concern with query access time of data segment only, and allocate the index segment into different channel. We assume to utilize Global Index, a multi channels indexing technique that we have proposed in [10]. The use of index in this scenario is simply to guide mobile client accessing data item of interest at the right moment, and channel. Figure 1(b) illustrates the situation when index segment and data segment are in separate channel. It should be noted that data item corresponds to database record or tuples, and data segment contains a set of data item. A complete broadcast file is referred as a broadcast cycle [7].

The rests of the section in this paper are organized as follows. Section 2 contains the related work of the proposed technique. It is then followed by the description of the proposed scheme and its application in section 3. Section 4 describes the simulation model and performance results of our proposed scheme as compared to conventional method. Finally, section 5 concludes the paper.

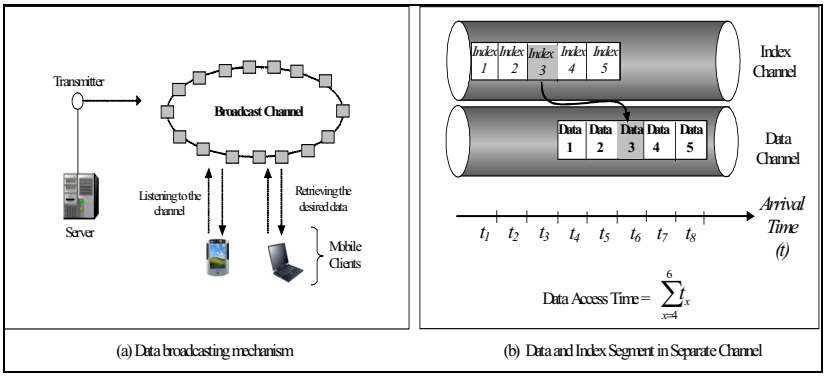

Fig. 1. Broadcast Mechanism with Index and Data Channel

2 Related Work

[2] presented a broadcast ordering scheme, where the hot items or the most frequently accessed data items are broadcast more often than cold items. The author allocates the hot items in the fastest bandwidth in decreasing order. The number of items in the channel is specified by given a temperature threshold value. [8] enhanced the work in [2] by considering different bandwidth posses by each channel and the size of each data items is varied. However, these approaches do not consider the relationship of one data item with the other. Therefore, these schemes are not effective for multiple data items retrieval.

[7] proposed a cost model called the Semantic Ordering Model (SOM) for identifying the optimal organization of database items over a broadcast channel. The cost model calculates the cost of each access graph, which is used to represent the dependency of data items. SOM is defined into two models namely Simple and Extended SOM. Simple SOM considers only the relationship among entity types while Extended SOM incorporates historical query access patterns. However, this technique concerns with single channel property only.

Our proposed technique considers single and multiple data items retrieval, and the data items are broadcast over multiple channel.

3 Broadcast Allocation for Multi Channel Data Dissemination: Proposed Scheme

3.1 Preliminaries

In this paper, we concern with request that returns multiple data items. A number of applications in this category includes a situation when mobile client wants to obtain more than one stock prices information at the same time (i.e. to list the stock price of all car companies under General Motors corp.) or when client wants to retrieve weather forecast cities concurrently (i.e. to list the weather forecast of all cities in a certain region). To accommodate this type of request, we need to consider the relationship between one data item and the others. Although, multiple data items retrieval is our primary objective, our proposed scheme also considers single data item retrieval. Thus, the application of this scheme is not limited to either case.

In order to improve the query access time, one needs to determine the broadcast order in advance. The broadcast order that we propose in this paper is designed to place the most requested data items close to each other in the channel. More importantly, we consider the access relationship between one item and another.

To find the query access information of mobile clients, we may incorporate either one of the following mechanism. One is to collect the access information from each mobile client at a regular interval, second is to determine the access information from the behaviour of offline or desktop users with the assumption that the same data items are accessible through the internet. Thus, we can assume that offline users and wireless users have a similar access patterns. Once, the access information is received by the server, the statistics will be compiled and the broadcast organisation scheme will be implemented. An example of query patterns is as follows:

$$Q1 = \{d_1, d_3\}$$
$$Q2 = \{d_1, d_3, d_4\}$$
$$Q3 = \{d_5\}$$
$$Q4 = \{d_6, d_7, d_8, d_9\}$$
$$Q5 = \{d_2, d_3, d_7\}$$

3.2 Broadcast Ordering Scheme

Let us denote $D = \{d_1, d_2, \ldots d_m\}$, which is a set of data items to be broadcast in the server, and Q as a set of queries $\{Q_1, Q_2, \ldots, Q_n\}$. In this case, we assume the data item has an equal size and the order of the retrieval can be arbitrary, which means if any of the required data by Q_i arrives in the channel, it will be retrieved first. Each query, Q_i, accesses a number of data items j, where $j \subset D$. The broadcast channel is indicated by C, and the length of the broadcast cycle in a channel is given by BC. We denote the broadcast schedule as $S = \{d_x, d_y, \ldots d_z\}$. Similarly, the broadcast program for each channel is defined by SC.

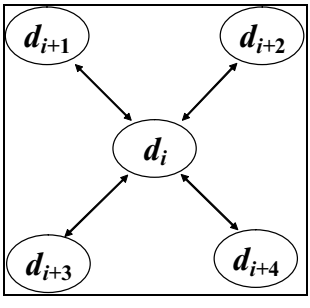

Fig. 2. The architecture of the proposed method

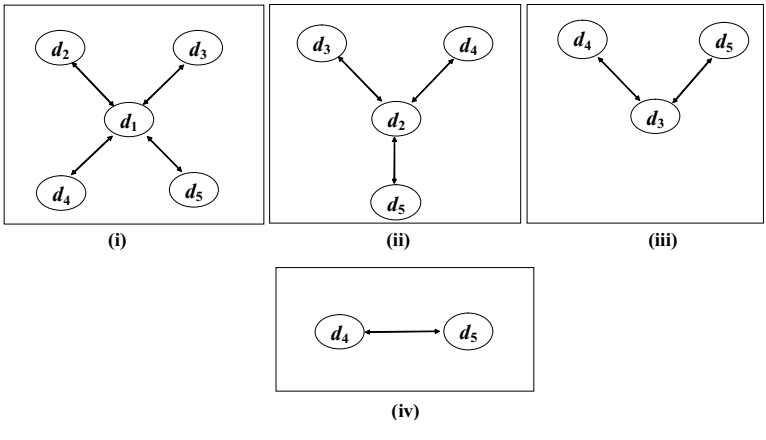

Fig. 3. An example of the proposed method process

The mechanism of our proposed scheme is divided into four stages.

- *Firstly*, we calculate the access frequency of each data item given by Q. Subsequently, the data items will be sorted in decreasing order of value of access frequency. The result will be temporarily stored in *ST1,* and incorporated into the final broadcast order S after the second stage is finalized. The algorithm of this stage is given in Figure 4.

Algorithm 1. Broadcast Ordering Algorithm – Stage 1

Input: An array of data items (*D),* queries (Q_i) with their related data items (j_i), and the total number of data items to be broadcast (*m).*
Output: An array of **b**roadcast program (*ST1*)

Procedure:
1. Initialise $i = 1$, $T = 0$
2. **Do while** $i \leq m$
3. **For** $n = 1$ to Q_{max}
4. Count the frequency of d_i in Q_n
5. Store and cumulate the result in temporary variable T
6. **If** Q_{max} then
7. $ST1_{(i)} = T$
8. Clear T
9. **End if**
10. **Next** n
11. increment i
12. **Loop**
13. Check the largest value in *ST1*
14. Ordered the value *ST1* such that $ST1\ (1) \geq ST1\ (2) \geq \ldots \geq ST1\ (m)$
15. **End Procedure**

Fig. 4. Stage 1 algorithm

- *Secondly*, the relationship between one data item and the other will be analyzed. The access frequency of each data item in relation with other data item will be assessed from given *Q*. This stage is required to allocate the related data items close to each other. Thus, the number of calculation required for this purpose will be $\frac{m \times (m-1)}{2}$, m is the total number of data items in *D*. Figure 2 illustrates this stage. It shows that each data item to be broadcast will be analyzed for its relationship with other data item based on *Q*. The calculation is done in sequence basis. *i* in Figure 2 reads the data item number. This process iterates until each data item has been counted. A single process includes two ways relationship of the data. Figure 3 depicts an example when there is 5 number of data items to be broadcast. Each data item will be counted for its relationship with the next sequence of the data item from given *Q*. The double point arrows indicate two ways relationship of the two data. In this case, $D = \{d_1, d_2, d_3, d_4, d_5\}$, we count the frequency of the two data items, which appear at the same time in a query. Started from d_1, which is counted towards the relation with the next sequences of the data item that is d_2, d_3, d_4, d_5.

This process continues sequentially until *m*-1 data item. After all data item has been analysed, they are sorted in non-increasing order based on the access frequency with other data item, and stored in *ST2*. Each allocation will involve two data items. If there is a duplication of data item within *ST2*, then data item with the highest *ST2* value are kept and the rests are removed. Figure 5 shows the algorithm of the second stage.

Algorithm 2. Broadcast Ordering Algorithm – Stage 2

Input: An array of data items (*D),* queries (Q_i) with their related data items (j_i), and the total number of data items (*m*).
Output: Array of broadcast program (*ST2*).

Procedure:
1. Initialise i = 1, *T* = 0
2. **Do while** *i* < *m*
3. **For** *k* = (*i* + 1) to *m*
4. **For** *n* = 1 to Q_{max}
5. Count the frequency of d_i retrieved at the same time with d_k in Q_n
6. Store and cumulate the result in temporary variable *T*
7. **If** Q_{max} then
8. *TDarray* (d_i, d_k) = *T*
9. Clear *T*
10. **End if**
11. **Next** *n*
12. **Next** *k*
13. increment *i*
14. **Loop**
15. Check the largest value in *TDarray*
16. Ordered the value in *TDarray* from the largest to the smallest value such that *TDarray* (1) ≥ *TDarray* (2) ≥ … ≥ *TDarray* (max)
17. The coordinate value for each *TDarray* determine the data item
18. Place the coordinate value for each *TDarray* to *ST2* with the non-increasing order. Check for any duplicate data item within *TDarray*. Data item can only appear one for each cycle. Thus, in case there is any duplication, data item with the highest *TDarray* value are kept and the rests are removed.
19. **End Procedure**

Fig. 5. Stage 2 algorithm

- The *third stage* is to combine *ST1* and *ST2* forming a single broadcast order *S*. We start with allocating the data item in *ST2* into *S* also in non-increasing order. This process continues until just one access frequency left to be processed. Subsequently, we take the result from the *ST1*, analyse it against *S*, the data item in *ST1* that has existed in *S* will be discarded. The left over data item is put into *S* with non-increasing order as well. The algorithm of this mechanism is given in Figure 6.

Algorithm 3. Broadcast Ordering Algorithm – Stage 3

Input: Two arrays of Broadcast schedule (*ST1 and ST2*)
Output: An array of Final Broadcast program (*S*)

Procedure:
1. **For** *i* = 1 to $ST2_{(max)}$
2. Move $ST2_{(i)}$ to *S*
3. **Next** *i*
4. **For** *i* = 1 to $ST1_{(max)}$
5. Check if $ST1_{(i)}$ exists in *S*
6. **If** $ST1_{(i)} \subset S$ then
7. skip
8. **Else**
9. Move $ST1_{(i)}$ to S_{max+1}
10. **End if**
11. **Next** *i*
12. **End Procedure**

Fig. 6. Stage 3 algorithm

- The *final stage* is to broadcast the data items over multiple channels. Figure 7 depicts the algorithm for allocating the broadcast order into multiple channels. We employ our previous work in [9] to determine the optimum number of data items in broadcast channel. The optimum number is indicated by *BC*. Once the optimum number is located, the number of broadcast channels should be increased. Subsequently, the length of broadcast cycle is split, and broadcast over multiple channels.

Algorithm 4. Multiple Channel Allocation

Input: An array of Broadcast schedule (*S*), optimum length of broadcast cycle in a channel (*BC*)
Output: Array of Broadcast program for each channel (*SC*)

Procedure:
1. Let channel number = x
2. Initialise x =1, i =1
3. **Do While** $i \leq S_{(max)}$
4. **If** $i \leq BC$
5. $SC(_x)= S(i)$
6. increment x
7. **End**
8. increment i
9. **Loop**
10. **End Procedure**

Fig. 7. Final stage algorithm

Having known the broadcast order enable us to determine the sequence of data items that minimise the average access time. The next issue to consider is to specify the broadcast program. The broadcast program indicates the schedule of the first item in *S* at the broadcast cycle, and the sequential items follow the order that has been generated. To achieve the most effective broadcast program, one must note the statistical patterns of users start listening or probing into the channel. In this paper, we assume the pattern follows the behaviour of normal distribution. We allocate the broadcast order *SC* into the program, where the first item in each *SC* is placed at point (μ + 0.6μ) in the channel. μ corresponds to mean statistical value of clients begin probing into a channel. The point is chosen to minimise the chance of clients missing the data of interest and have to wait for the subsequent cycle. With this way, majority of requests are served within a single broadcast cycle. Although this means about 50% of requests have to wait for a short time period but the overall performance will be better since only a fraction of requests need to wait for the next cycle. The broadcast program can be generated at a regular interval.

4 Performance Evaluation

In this section, we analyze the performance of our proposed method as compared to conventional method. We relate conventional method with an arbitrary method when the data items are broadcast arbitrarily. To determine the optimum number of items in

a channel, we can use our technique in [9]. Thus, we can say that this paper is a subsequent stage of our previous work to further improve the query access time. In our previous work, the data are broadcast without any specific order.

To evaluate our scheme, we develop some simulation models. The simulation is carried out using two software packages namely Visual Basic 6.0 and *Planimate* or animated planning platforms [6]. The algorithm to determine the best broadcast order is coded in Visual Basic 6.0. VB 6 is preferable for us since we have utilized this software to build a real application in wireless environment. The application will be used to verify the result of the proposed method. The prototype of our system has been reported in [11]. The access time performance is done using *Planimate* simulation tool. The channel bandwidth is specified based on the EDGE standard.

Table1. Parameters of Concern

Parameters	Value
Size of each data Item	2KB
Number of Data Items	60
Bandwidth	64Kbps
Query Patterns/Profiles	10
Number of Dependent Items in Query	1-4
Number of Broadcast Channel	3
Number of Request	30

The simulation environment is set to apply exponential distribution for data item inter-arrival rate given an average value. We run the simulation for thirty iterations, and derive the average result accordingly. In the query profile, we consider request that return one and more number of items. We determine a range parameter of number of dependent items in query. Subsequently, we analyze both situations when the access frequencies of each query are uniform and varied. The parameters of concern are given in Table 1. We can see from Figure 8 our proposed scheme outperforms the conventional method with about two time lower average access time. We also vary the access frequencies of each query pattern, and it results a slightly lower access time as compared to the uniform pattern frequencies.

In Figure 9, we modify the number of data items to be broadcast. It is shown, that the increase of broadcast items severely affects the conventional method. We increase the data items twice as the number of data items in Figure 8, and the result for conventional one indicates the access time improves twice as much. In the contrary, our proposed scheme increase less than twice or can be considered as minor increase. Consequently, the gap between our proposed method and conventional method is larger. The reason for these two cases is that since we allocate the most requested items close to each other, the increase of data items would not affect much of the overall query performance. Furthermore, with the same items requested many times, the better the performance will be.

Note that we have the background to determine the number of channel required to obtain optimum result.

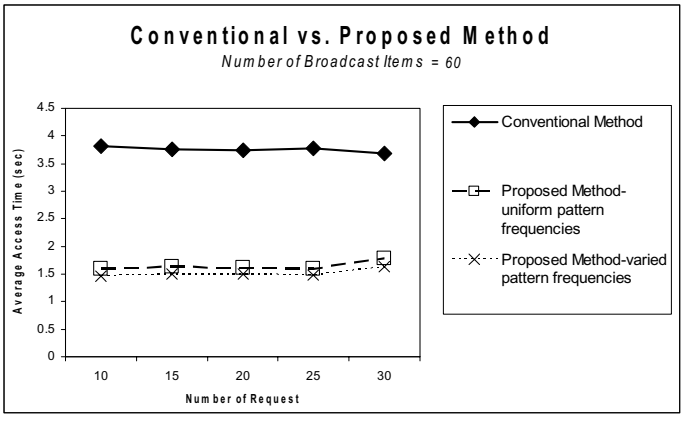

Fig. 8. Conventional vs. Proposed Method (60 broadcast data items)

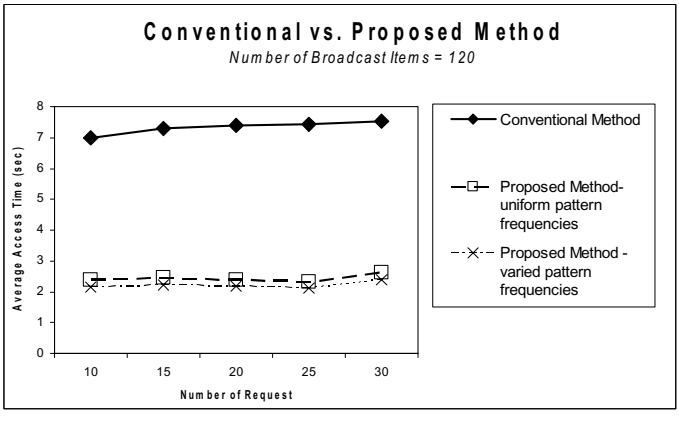

Fig. 9. Conventional vs. Proposed Method (120 broadcast data items)

5 Conclusions and Future Work

In this paper, we have presented a scheme to order the data items in the broadcast channel in a way that most clients will have a minimum query access time. This scheme is a continuation of our previous work which concern with multiple channel environments. We have developed some simulation models to analyze the performance of our proposed scheme. We found that our scheme provide a substantially better average query access time as compared to conventional method.

For future work, we will incorporate data replication in the broadcast program. In order to obtain an effective result, we need to investigate the degree of the replication as well as the structure of the broadcast program.

References

1. Barbara D., "Mobile Computing and Databases-A Survey", *IEEE TKDE*, **11**(1):108-117, 1999.
2. Prabhakara K., Hua K.A, and Jiang N., "Multi-Level Multi-Channel Air Cache Designs for Broadcasting in a Mobile Environment", In *Proc. of the IEEE ICDE*, pp. 167-176, 2000.
3. Malladi R. and Davis K.C., "Applying Multiple Query Optimization in Mobile Databases", In *Proc. of the 36th Hawaii International Conference on System Sciences*, pp. 294 – 303, 2002.
4. Myers B.A. and Beigl M., "Handheld Computing", IEEE Computer Magazine, **36**(9):27-29, 2003.
5. Paulson L.D.,"Will Fuel Cells Replace Batteries in Mobile Devices?" *IEEE Computer Magazine*, **36**(11):10-12, 2003.
6. Seeley D. et al, *Planimate™-Animated Planning Platforms*, InterDynamics Pty Ltd, 1997.
7. Si A. and Leong H. V., "Query Optimization for Broadcast Database", *DKE,* **29**(3): 351-380, 1999.
8. Tran D.A., Hua K.A, and Jiang N., "A Generalized Design for Broadcasting on Multiple Physical-Channel Air-Cache", In *Proc. of the ACM SIGAPP*, pp. 387-392, 2001..
9. Waluyo, A.B., Srinivasan B., Taniar D., "Optimal Broadcast Channel for Data Dissemination in Mobile Database Environment", In *Proc. of the APPT'03*, LNCS, **2834**: 655-664, 2003a.
10. Waluyo, A.B., Srinivasan B., Taniar D., "Global Index for Multi Channels Data Dissemination in Mobile Databases", In *Proc. of the ISCIS'03,* LNCS, Springer-Verlag, **2869**:210-217, 2003b.
11. Waluyo A.B., Hsieh R., Taniar D., Rahayu W., and Srinivasan B., "Utilising Push and Pull Mechanism In Wireless E-Health Environment", In *Proc. of the IEEE EEE'04*, pp. 271-274, 2004.
12. Xu J., Zheng B., Lee W-C., and Lee D.L., "Energy Efficient Index for Querying Location-Dependent Data in Mobile Broadcast Environments", In *Proc. of ACM SAC'03*, pp. 239-250, 2003.
13. Yajima E., Hara T., Tsukamoto M. and Nishio S. "Scheduling and Caching Strategies for Correlated Data in Push-based Information Systems", *ACM SIGAPP*, **9**(1): 22-28, 2001.