Index and Data Allocation on Multiple Broadcast Channels Considering Data Access Frequencies

Chih-Hao Hsu, Guanling Lee and Arbee L.P. Chen
Department of Computer Science and Information Engineering
National Dong Hwa University
Hualien, Taiwan 973, R.O.C.
Email: alpchen@cs.nthu.edu.tw

Abstract

In a wireless environment, the bandwidth of the channels and the energy of the portable devices are limited. Data broadcast has become an excellent method for efficient data dissemination. In this paper, the problem for generating a broadcast program of a set of data items with the associated access frequencies on multiple channels is explored. In our approach, we consider allocating index information and data items on multiple broadcast channels by extending the distributed indexing approach. Moreover, global data replication and local data allocation are performed to improve the average access time of all data items. Simulation is performed to compare the performance of our approach with an existing approach. The result of the experiments shows that our approach outperforms the existing approach.

Keywords: Wireless Environment, Multiple Broadcast Channels, Data Allocation, Broadcast Program, Index Information.

1. Introduction

With the development of wireless technologies, people can now access information any time, any where via wireless communications. However, different from traditional wired networks, some issues should be considered in the wireless environment. First, the bandwidth of the wireless network and the energy needed for portable devices are limited. Second, the environment is asymmetry, that is, from the power consumption point of view, sending data is more costly than receiving data for a portable computer. Due to these issues, the traditional request-response system [21] is no longer suitable for data dissemination in the wireless environment. Therefore, data dissemination in the wireless environment has become an interesting research problem [6][14][19].

Broadcast-based information systems provide the dissemination of information with a cost independent of the number of clients, which compensates for the limited bandwidth in the wireless environment. Moreover, the clients can retrieve the broadcast data by just tuning to the broadcast channel, which results in a certain degree of energy saving. Therefore, data broadcast has become an attractive solution for information dissemination. In the broadcast-based system, the clients have to access data items in the broadcast channel sequentially. How to allocate data items in the broadcast channel for efficient data access becomes an important issue.

In order to generate *broadcast program* for data dissemination, there are two factors to consider:

- ***Access time***. It is the time elapsed from the moment an initial probe is made into the broadcast channel to the moment the desired data are acquired. This is the total time the clients must spend and is often used to evaluate the performance of the broadcast programs.
- ***Tuning tim*e**. It is the time spent by the clients listening to the broadcast channel. There are two modes for the clients to operate. When the clients are listening to the data items in the broadcast channel, the CPU must operate in the *active mode*, which is costly for power consumption. However, the clients can operate in the *doze mode* to save power consumption, when the requested data items have not arrived.

In the broadcast-based system, a broadcast program needs to be constructed to determine the order of data items to be broadcast. The main issue to generate a broadcast program is to minimize the average access time for saving the bandwidth and energy in a mobile computing system. Many researcheres have focused on generating broadcast programs for a single broadcast channel. However, few researcheres attempted to minimize the access time and tuning time on the multiple channels environment.

In this paper, the method of data and index allocation on multiple broadcast channels is proposed. We devote ourselves to minimizing the average access time and tuning time of the broadcast program. The rest of this paper is organized as follows. Section 2 introduces the related work in this domain. In Section 3, the technique for allocating data and index on the multiple channels is proposed. The performance analyses are studied in Section 4. Finally, in Section 5, conclusion and future work are presented.

2. Related Work

In [BGH92][HGL87], the server uniformly broadcasts each requested data item. However, in fact, some data items are more frequently accessed than others. Acharya et al. [1] propose the concept of broadcast disks, in which all data items are partitioned into several groups such that the groups containing data items with higher access frequencies have shorter broadcast periods. As a result, the average access time decreases. The performance of broadcast disks is further improved in [2][3][4]. Moreover, approaches considering broadcasting variable-sized data items are proposed in [13][23]. However, these techniques do not consider generating indices to reduce the tuning time.

Imielinski et al. propose several indexing techniques for accessing data items on the broadcast channel. In *(1,m)* indexing [15], the index information is broadcast m times for each broadcast period. *Distributed indexing* [15] improves it by only replicating the index information partially. Another technique called *flexible indexing* was proposed in [16]. In flexible indexing, the broadcast program is divided into several equal-sized segments. In each segment, some index information is provided to navigate all data items on the broadcast channel. However, these techniques assume that the access frequencies of all data items are the same. Chen, Yu and Wu [9] propose an algorithm, called *CF* (constant fanout), to generate an imbalanced index tree for skewed data accesses. The basic concept of CF is to make the data item with high access frequency to approach the root node so that the access time can be minimized. However, all of these techniques only apply to the *flat broadcast program*. In a non-flat broadcast program, data items with high access frequency are broadcast more frequently. Some techniques are proposed in [22][24] to solve this problem. These techniques generate broadcast programs using the broadcast disks method so that the broadcast program contains several segments. Each segment can be treated as a flat broadcast program and some index information can be attached to it. All of these techniques reduce tuning time by skipping irrelevant data items.

There are some other works focusing on generating broadcast programs on multiple channels. Peng and Chen ([18]) transform the problem of generating broadcast programs on multiple channels into one of constructing a channel allocation tree with variant fan-out and develop a heuristic algorithm to minimize the average access time of a broadcast program on multiple channels. However, this technique does not consider how to construct index structures. In [HT71], a binary search tree, called *Alphabetic Huffman Tree* is proposed. Shivakumar et al. [20] extend it to a k-nary search tree and allocate this search tree to multiple channels. However, it is inflexible because the number of channels must equal to the height of the tree. Lo and Chen [17] propose a solution for optimal index and data allocation, which minimizes the average access time for any number of broadcast channels. Moreover, in [10], the issue of allocating dependent data on multiple channels is discussed. A heuristic algorithm is proposed to cluster related data items to minimize the average access time.

3. The Multiple Broadcast Segment-Based Method

3.1 Preliminary Concepts

We extend the work of *distributed indexing* proposed in [15] to the multiple channels environment and consider the problem of data replication. As a result, the access time and tuning time of the broadcast program are minimized.

Distributed indexing was proposed in [15]. There are three different index distribution methods, i.e., non-replicated distribution, entire path replication and partial path replication, differing in the degree of the replication of the index information. We adopt the entire path replication method (abbreviated *EPR*) in our approach. In EPR, all data items are associated with an index tree (see Figure 1). In this index tree, all data items are allocated in the leaves and the index nodes are built above these data items. The index tree consists of two parts, *replicated index part* and *non-replicated index part*. The top r levels of the index tree are the replicated index part, while the other levels are non-replicated index part. The index nodes of the (r + 1)$_{th}$ level are called non-replicated roots (abbreviated *NRR*). Each index sub-tree rooted in an index node in NRR appears only once in a broadcast cycle. However, each index node in the replicated index part appears more than once in a broadcast cycle. In the following, we will show how to generate a broadcast program based on this index tree.

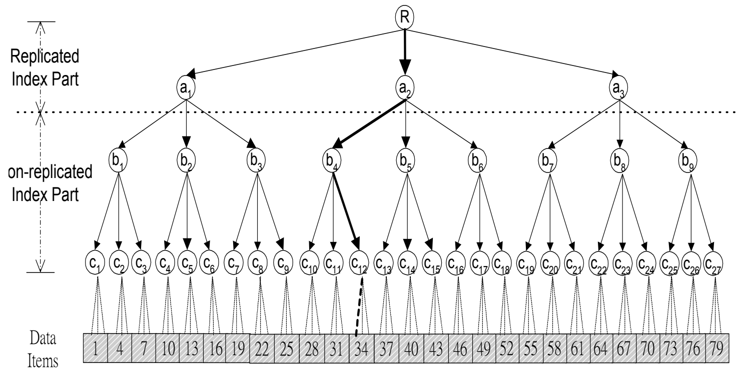


Figure 1: Data items associated with an index tree

Definition

B_i: The *i*th index node in NRR.

Rep(B_i): The sequence of index nodes along the path from the root of the index tree to the non-replicated root B_i (excluding B_i).

Ind(B_i): The index nodes of the index tree rooted at B_i.

Data(B_i): The set of data items indexed by B_i.

Let $NRR = \{B_1, B_2, \ldots, B_t\}$. The broadcast program for the index tree is a sequence of triples:

$< Rep(B_i), Ind(B_i), Data(B_i) > \quad \forall\ B_i \in NRR$,

in the left to right order.

The resultant broadcast program for the index tree is shown in Figure 2, where each triple forms a *broadcast segment,* denoted by *BS*, and the BS containing the triple $< Rep(B_i), Ind(B_i), Data(B_i) >$ is called BS(i).

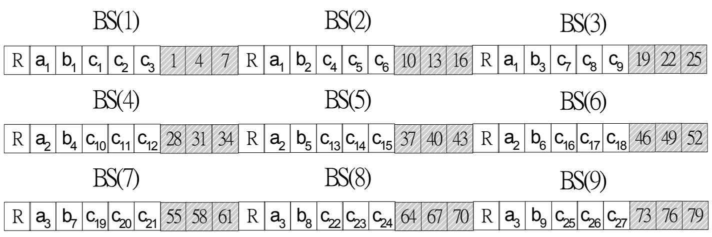

BS(1)
BS(2)
BS(3)
R a1 b1 c1 c2 c3 1 4 7 R a1 b2 c4 c5 c6 10 13 16 R a1 b3 c7 c8 c9 19 22 25
BS(4)
BS(5)
BS(6)
R a2 b4 c10 c11 c12 28 31 34 R a2 b5 c13 c14 c15 37 40 43 R a2 b6 c16 c17 c18 46 49 52
BS(7)
BS(8)
BS(9)
R a3 b7 c19 c20 c21 55 58 61 R a3 b8 c22 c23 c24 64 67 70 R a3 b9 c25 c26 c27 73 76 79

Figure 2: The resultant broadcast program of the index tree

In order to quickly find out the desired data items, each node in the broadcast contains some indexing information. All nodes (index nodes and data items) have a pointer to the beginning of the next BS so that a client can probe to the root of the index tree and trace the index information held in the index nodes to traverse the index tree. Besides, the first node in each BS contains a pointer to the beginning of the next broadcast cycle so that a client can check whether the desired data item has been missed and goes to the next broadcast cycle directly. According to the index information, a client can search for a data item with key *k* as follows:

- Tune to the current node and get the offset to the next BS, and doze off until the beginning of the next BS.
- Check whether the client has missed the desired data item, if that is the case, doze off until the beginning of the next broadcast cycle.
- Trace the index tree and find out the desired data item.

We use an example to illustrate the above steps. Refer to Figure 2, assume the first probe for a client is in node 64 and the desired data item is in node 34. Then, the client makes the following probes *64*, *ninth_R*, *first_R*, *first_*a_2, b_4, c_{12}, and *34* to get the desired data item.

3.2 Extend Distributed Indexing into Multiple Channels

In the distributed indexing method, the access frequencies of data items are not considered. However, in most situations, some data items are accessed more frequently than others. Considering the access frequencies of data items to generate a broadcast program will minimize the average access time and the tuning time for all data items. In our approach, we will take the access frequencies of data items into consideration to generate an *initial broadcast program*, followed by a refinement of the initial broadcast program.

3.2.1 Generating initial broadcast programs

In the distributed indexing, a sequence of triples $< Rep(B_i), Ind(B_i), Data(B_i) >$ is formed. Assume that the summation of the access frequencies for the data items involved in $Data(B_i)$ is $Sum(B_i)$, our approach for generating initial broadcast programs works as follows:

- Sort the sequence of triples $< Rep(B_i), Ind(B_i), Data(B_i) >$ in descending order according to the value of $Sum(B_i)$.
- Allocate a triple in the order to broadcast in each channel.
- Repeat the allocation until all triples are broadcast in the channels.

Figure 3 illustrates our approach using the same data set shown in Figure 1. In this example, assume the number of channels is 3 and the summation of access frequencies for each BS(i) is BS(1) > BS(4) > BS(7) > BS(2) > BS(5) > BS(8) > BS(3) > BS(6) > BS(9). First, BS(1), BS(4) and BS(7) are selected to be broadcast in each channel. This forms a *multiple broadcast segment*, denoted by *MBS*. Moreover, MBS(i) is used to denote the ith MBS in the broadcast channels. After that, BS(2), BS(5) and BS(8) are selected to be broadcast. Finally, the BS(3), BS(6) and BS(9) are selected to be broadcast. In the following, we will consider the problem of MBS replication and the problem of data allocation in an *MBS* to refine the initial broadcast program.

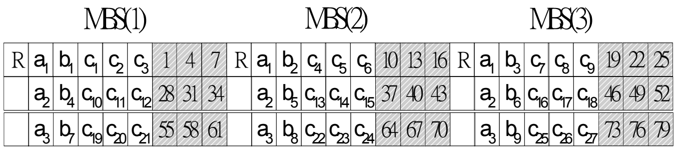

MBS(1)
MBS(2)
MBS(3)
R a1 b1 c1 c2 c3 1 4 7 R a1 b2 c4 c5 c6 10 13 16 R a1 b3 c7 c8 c9 19 22 25
a2 b4 c10 c11 c12 28 31 34 a2 b5 c13 c14 c15 37 40 43 a2 b6 c16 c17 c18 46 49 52
a3 b7 c19 c20 c21 55 58 61 a3 b8 c22 c23 c24 64 67 70 a3 b9 c25 c26 c27 73 76 79

Figure 3: The resultant initial broadcast program in multiple channels

3.2.2 Refining initial broadcast programs

Replication of MBSs

As mentioned, in practice, some data items are accessed more frequently than others. Therefore, it makes sense to broadcast data items with higher access frequencies more frequently. In the multiple broadcast channels, the summation of access frequencies for the data items in the multiple broadcast segments *MBS*(*i*), denoted by *M_SUM*(*i*), is $\sum_{BS(j)\in MBS(i)} Sum(j)$. According to the property shown in [25], for fixed sized data items, the total average access time will be minimized if the instances of each data item are equally spaced and for any two data items d_i and d_j, $p_i / p_j = \sqrt{f_i} / \sqrt{f_j}$, where p_i is the reciprocal of s_i which is the probability that d_i will be selected to broadcast in each time slot, and f_i is the access frequency of d_i. Assume that the broadcast channels are divided into segments with the size of an *MBS*. We can replicate the *MBS*s as follows:

```
Algorithm group_data_replication:
Input: The set of multiple broadcast segments MBSs = {MBS(1), MBS(2), …, MBS(k)} and the corresponding summation of access frequencies M_SUM(i).
Output: A broadcast program.
Begin
1. Let the $r_k$ (the number of replication) for the MBS(k) with the smallest M_SUM(k) be 1. Decide integer $r_i$ for each MBS(i) such that $r_i$ is as close to $\sqrt{M\_SUM(i)} / \sqrt{M\_SUM(k)}$ as possible for all i, i≠k.
2. for i = 1 to N
      Allocate the copies of MBS(i) to the broadcast channels such that the distance between any two copies of MBS(i) is as close to $\frac{\sum_{j=1}^{N} r_j}{r_i}$ as possible.
End
```

Refer to Figure 3, assume that the summation of access frequencies in *MBS*(1) is 0.5 and the summation of access frequencies in *MBS*(2) and *MBS*(3) are 0.25. According to the property shown in [25], We will broadcast *MBS*(1) twice in a broadcast cycle and broadcast *MBS*(2) and *MBS*(3) once in a broadcast cycle. The resultant broadcast program is shown in Figure 4.

R	a_1	b_1	c_1	c_2	c_3	1	4	7	R	a_1	b_2	c_4	c_5	c_6	10	13	16
	a_2	b_4	c_{10}	c_{11}	c_{12}	28	31	34		a_2	b_5	c_{13}	c_{14}	c_{15}	37	40	43
	a_3	b_7	c_{19}	c_{20}	c_{21}	55	58	61		a_3	b_8	c_{22}	c_{23}	c_{24}	64	67	70

R	a_1	b_1	c_1	c_2	c_3	1	4	7	R	a_1	b_3	c_7	c_8	c_9	19	22	25
	a_2	b_4	c_{10}	c_{11}	c_{12}	28	31	34		a_2	b_6	c_{16}	c_{17}	c_{18}	46	49	52
	a_3	b_7	c_{19}	c_{20}	c_{21}	55	58	61		a_3	b_9	c_{25}	c_{26}	c_{27}	73	76	79

Figure 4: The resultant broadcast program after global data replication

Data Allocation in an MBS

In order to further reduce the average access time for all data items, we consider data allocation in an *MBS*.

As mentioned in Section 3.1, *BS*(*i*) consists of three parts, *Rep*(B_i), *Ind*(B_i) and *Data*(B_i). According to the index tree shown in Figure 1, we have to broadcast the part of *Rep*(B_i) first such that the clients can trace the index tree from the root. After, we have to consider the allocation of the index nodes in *Ind*(B_i) and data items in *Data*(B_i). As shown in Figure 5, assume that the fan-out of the index tree is *K*, *Ind*(B_i) consists of K sub-trees. In each sub-tree, the summation of the access frequencies of the data items contained in this sub-tree is calculated and all sub-trees are sorted in descending order accordingly. The sub-trees with higher summations of access frequencies will be allocated first such that the average access time can be reduced. Recursively, Each sub-tree also consists of K sub-trees. We can allocate these sub-trees in the same way.

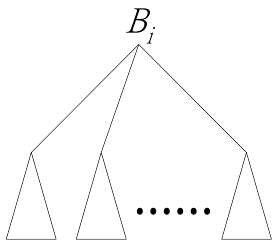


Figure 5: The structure of Ind(B_i)

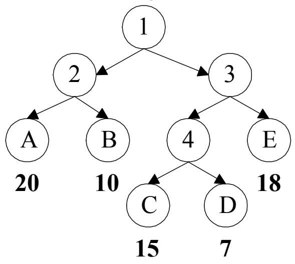
The following example is used to illustrate our approach. Refer to Figure 1, assume the summations of access frequencies of data items indexed by c_1, c_2 and c_3 are 0.25, 0.15 and 0.2 respectively. The allocation of index nodes and data items for the triple < *Rep*(B_1), *Ind*(B_1), *Data*(B_1) > is shown in Figure 6.

R	a_1	b_1	c_1	1	c_3	7	c_2	4

Figure 6: The allocation of the triple <Rep(B_1), Ind(B_1), Data(B_1)>

4. Performance Evaluation

In order to evaluate the performance of the proposed algorithm, a set of simulations is performed by generating different broadcast data sets. In the simulation, the cost metric is the average access time and the tuning time of all data items. We compare the cost of our approach with that of an existing algorithm proposed in [17]. In [17], this problem is transformed to the *Directed Optimal Linear Ordering Problem*. Moreover, a polynomial time algorithm is proposed to solve this problem in a single broadcast channel environment. After that, this paper provides a linear time algorithm to transform the result to the multiple channels. This is shown in Figure 7.



C_1	1	2	A	B	3	E	4	C	D

C_1	1	2	A	E	C	1	2	A	E
C_2		3	B	4	D		3	B	4

(a) Index tree (b) One channel (c) Two channels

Figure 7: The method of index and data allocation in [17]

4.1 Simulation Model

The following parameter is used to generate different broadcast data sets.

PARAMETERS

- *N*: The number of data items being broadcast.
- *θ*: The parameter of Zipf distribution.
- *Data size*: the size of a data item
- *Index size*: the size of an index node

Parameters	Default value	Ranges
Number of data items (*N*)	1000	27 – 2187
Zip parameter (*θ*)	0.5	0 – 0.99
Ratio (Data size/Index size)	1	1-5

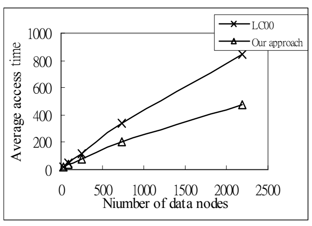
Table 1: Parameter Settings

The parameter settings for our experiments are listed in Table 1. The access frequencies of data items are based on the Zipf distribution [GSE94]. In the Zipf distribution, the access frequencies for the data items follow the 80/20 rule that 80 percent clients are usually interested in 20 percent data items.

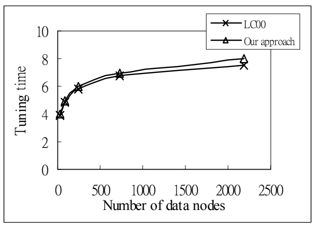
4.2 Performance Evaluation

4.2.1 Effect of the Number of Nodes

In this simulation, the effect of the number of channels is considered. The result is shown in Figure 8. As shown in Figure 8(a), the average access time of all data items increases as the number of the channels increases. Intuitively, as the number of channel increases, the length of the broadcast cycle increases. Therefore, the average access time of all data items increases. In this experiment, we find out that our approach outperforms the other for the average access time. The reason is that our approach replicates the data items with high access frequencies on the broadcast channels such that the average access time of these data items is reduced. Also, the average tuning time is shown in Figure 8(b). We can see that the average tuning time for the two approaches is almost the same.



(a)The average access time

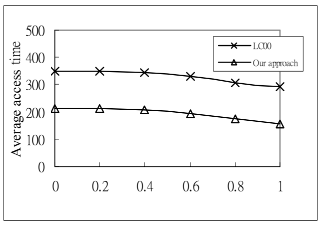


(b) The tuning time

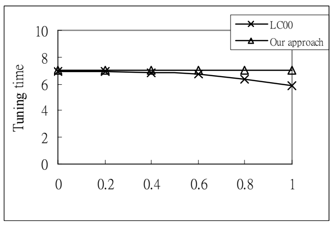
Figure 8: Effect of the number of nodes

4.2.2 Effect of Zipf Parameter θ

Another factor that affects the performance of the broadcast program is the access distribution of the data items. In this simulation, the access frequencies of data items is based on the Zipf distribution. In the Zipf distribution, as the Zipf parameter θ tends to 1, the access frequencies of data items become skew, that is, few data items have high access frequencies. In this experiment, we show the effect of different skew degrees of access frequencies. The experiment result is shown in Figure 9. When the access frequencies become skewer, a small number of data items are accessed more frequently. Therefore, our approach reduces the average access time of these data items by replicating these data items more frequently. This confirms with the result shown in Figure 9(a). Figure 9(b) shows the average tuning time. Because the index tree proposed in [17] is a skew index tree, the average tuning time decreases in this approach as θ tends to 1.



(a)The average access time

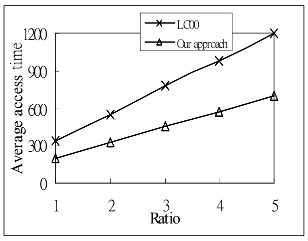


(b) The tuning time

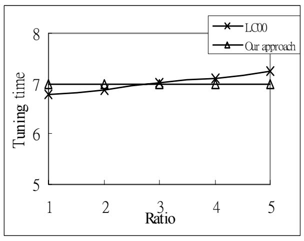
Figure 9: Effect of Zipf Parameter θ

4.2.3 Effect of the Ratio of the Size of Data Item to the Size of Index Node

In our approach, the data items and index nodes are allocated in different time slots. However, the approach proposed in [17] mixes the data items and index nodes in the same time slot. In [17], as the size of data items is greater than the size of index nodes, some broadcast bandwidth allocated to index nodes is wasteful. In this experiment, we show the effect of the ratio of the size of data item to the size of index node. The experiment result is shown in Figure 10. Our approach outperforms the other on the average access time and the average tuning time as the size of data item becomes larger than the size of an index node.



(a)The average access time



(b) The tuning time

Figure 10: Effect of the ratio of the size of data item to the size of index node

5. Conclusion

In this paper, a heuristic algorithm of allocating index information and data items on the multiple channels is proposed. In our approach, the distributing indexing method is extended to the multiple channels and the access frequencies of data items are considered. Moreover, data replication between *MBS*s and data allocation in an *MBS* are presented to reduce the average access time of all data items. Simulation is performed to compare the performance between our approach with an existing approach. The result of experiment shows that our approach is better than the other approach.

There are many applications that allow the clients to access multiple data items at a time. How to allocate all data items on multiple channels to minimize the average access time and the tuning time is a challenge. Moreover, how to generate the broadcast program to adapt to the changing access frequencies is also a problem to solve.

Reference

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast Disks: Data management for Asymmetric Communication Environments," *Proc. ACM SIGMOD Conf.*, pages. 199-210, San Jose, CA, May 1995.

[2] S. Acharya, M. Franklin and S. Zdonik, "Dissemination-based Data Delivery Using Broadcast Disks", *IEEE Personal Communications*, 2(6), Dec. 1995.

[3] S. Acharya, M. Franklin and S. Zdonik, "Disseminating Updates on Broadcast Disks", *Proc. VLDB Conference*, pages 354~365, 1996.

[4] S. Acharya, M. Franklin and S. Zdonik, "Prefetching from a Broadcast Disk", *Proc. IEEE International Conference on Data Engineering*, pages 276~285, 1996.

[5] S. Acharya, M. Franklin and S. Zdonik, "Balancing Push and Pull for Data Broadcast", *Proc. ACM SIGMOD Conference*, pages 183~194, 1997.

[6] R. Alonso and H. Korth. "Database Systems in Nomadic Computing" In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pages 388-392, 1993

[7] Y. D. Chung and M. H. Kim, "QEM: A Scheduling Method for Wireless Broadcast Data", *Proc. International Conference on Database Systems for Advanced Applications proceedings*, pages 135~142, 1999.

[8] J. Cai and K.L. Tan. "Tuning Integrated Disseminated-based Information Systems". *Data an Knowledge Engineering*, 30(1): pages 1-21, 1999.

[9] M. S. Chen, P. S. Yu and K. L. Wu, "Indexed Sequential Data Broadcasting in Wireless Mobile Computing", *Proc. IEEE International Conference on Distributed Computing System*s, pages 124~131, 1997.

[10] A.R. Hurson, Y.C. Chehadeh and J. Hannan, "Object Organization on Parallel Broadcast Channels in a Global Information Sharing Environment," *IEEE International Performance, Computing, and Communications Conference (IPCCC)*, February 2000.

[11] Quinlong Hu, Dik Lun Lee, Wang-Chien Lee "Optimal Channel Allocation for Data Dissemination in Mobile Computing Environments". *In Proceeding of the 1998 International Conference on Distributing System*, pages 480-487, 1998.

[12] S. hameed and N. Vaidya. "Efficient Algorithms for Scheduling Data Broadcastation"

[13] S. hameed and N. Vaidya. "Efficient Algorithms for Scheduling Data Broadcastation". *ACM/Baltzer Wireless Networks*, 5(3):183-193,1999.

[14] T. Imielinski, B.R. Badrinath, "Data Management for Mobile Computing" *SIGMOD RECORD*, 22(1): 34-39, 1993

[15] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Energy Efficient Indexing on Air," *Proc. ACM SIGMOD Conf.*, pages 25-36, 1994.

[16] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Power Efficient Filtering of Data on Air," *4th International Conference on Extending Database Technology (EDBT)*, pages 245-258, 1994.

[17] S.C. Lo and A. L. P. Chen. "Optimal Index and Data Allocation in Multiple Broadcast Channels" *In Proceeding of the 16th International Conference on Data Engineering*, pages 293-302, 2000

[18] W.C. Peng and M.S. Chen, "Dynamic Generation of Data Broadcast Programs for a Broadcast Disk Array in a Mobile Computing Environment" *Proc. of the ACM 9th Intern'l Conf. on Information and Knowledge Management*, November 6-11, 2000.

[19] E. Pitoura and G. Samaras "Data Management for Mobile Computing", *Kluwer Academic Publishers*, 1998.

[20] N. Shivakumar and S. Venkatasubramanian, "Energy-Efficient Indexing For Information Dissemination In Wireless Systems," *ACM, Journal of Wireless and Nomadic Application*, 1996.

[21] K.L Tan and B.C. Ooi. "Batch Scheduling for Demand-driven Servers in Wireless Environment"

[22] K.L Tan and J.X. Yu, "Energy Efficient Filtering of Non-uniform Broadcast", *In Proceeding of the 1996 International Conference on Distributing System*, pages 520-527, 1996.

[23] N. Vaidya and S. Hameed. "Scheduling data broadcast in asymmetric communication environments" *ACM/Baltzer Wireless Networks*, 5(3):171-182,1999.

[24] J.X. Yu and K.L Tan, "An Analysis of Selective Tuning Schemes for Non-uniform Broadcast", *Data and Knowledge Engineering*, 22(3): 319-344, 1997

[25] J.W. Wong. "Broadcast delivery", *Proceeding of the IEEE*, 76(12): 1566-1577, 1988