A Broadcast Disk Scheme for Mobile Information System

Putra Sumari, Rozaida Mat Darus and Amir Rizaan Rahiman
School of Computer Sciences, Universiti Sains Malaysia,
Minden, 11800, Penang, Malaysia

ABSTRACT

The rapidly expanding technology of cellular communications, wireless local area network (LAN), wireless data networks and satellite gives mobile users the ability of accessing information anywhere and anytime. Data broadcasting algorithm plays an important role as it instructs the server to disseminate large amount of data at a fast rate to many mobile clients. One of the work known as the Broadcast Disk method creates a sequence of slots containing data and broadcasts them in a circular manner on the air. Mobile clients keep listening to the air and catch those slots that interest them. However in the Broadcast Disk method, some slots are empty, which results to waste of resources and increase in clients' average waiting time. Empty slots are randomly generated and difficult to be located. In this paper we present a scheme called the Optimum Broadcast Disk, which is an enhancement of the Broadcast Disk method to overcome the empty slots issue and hence reduce the users' waiting time. We manage to locate these empty slots and fill them with popular data. We demonstrate by case studies and simulation that our scheme eliminates empty slots and minimizes clients' average waiting time.

Keywords: Data broadcasting, Broadcast Disk, Broadcast Schedule, Latency, Mobile Information System.

1. INTRODUCTION

The emergence of powerful portable computers, along with advances in wireless communication technologies, has made mobile computing a reality. In the evolving field of mobile computing, there is a growing concern on providing the mobile users the timely access to a large amount of information [1 – 3] which is known as mobile information system. Some examples of the existing mobile information system applications are whether broadcast system, highway condition monitoring system, traffic direction system as well as news and stock queries information system [2], [4]. Fig. 1 illustrates a scenario of the mobile information system. The system consists of server, transmitter, broadcast channel and mobile clients. The server stores data and broadcasts them to mobile clients through the transmitter. The transmitter transforms data into signals and sends them to the broadcast channels. Mobile clients then download the data that they are interested from the broadcast channel.

A well-known technology in broadcasting strategy is the asymmetric type of communication. The bandwidth at the downstream (server to clients) is much greater than that at the upstream (client to server) direction. It is a one way direction of data flow from the server to the mobile clients. A well-known asymmetric data broadcasting type is the Broadcasting Disk (BD) method [5 – 7]. The server continuously and repeatedly broadcasts data to the mobile clients. The server creates a series of slots which are filled with data and broadcasts them in a circular manner to the broadcast channel. Mobile clients keep listening to the channel and catch those slots that interest them. The advantages [5], [13] of having cyclic slots of broadcasting in asymmetric fashion are: (1) scalability, where the server simply accumulates all the users' requests that were made at the same slots and broadcasts those slots cyclically to the users. (2) It allows the server to broadcast different slots with different frequencies, where the more popular slots are broadcasted more frequently compared to those less popular slots. (3) Saves on power consumption of the mobile clients' device by allowing the server to do the entire transmitting job, leaving the mobile clients to just pick any data that interest them.

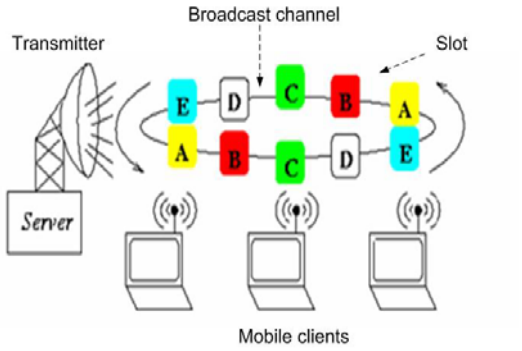


Fig. 1: Mobile information system.

One weaknesses of the Broadcast Disk (BD) method is the empty slots issue [5]. The BD algorithm produces empty slots in random a manner and they are difficult to locate. In this paper we propose an Optimum Broadcast Disk scheme, which is an enhancement of the BD algorithm, to eliminate the empty slot issue. In our approach, we manage to locate the empty slots and fill them with popular data. We decide not to delete them; instead we fill them with data. By doing so, we can maintain the original structure of series of slots produced by the BD algorithm. Our approach shows that the average waiting time of the mobile clients are reduced.

The rest of the paper is organized as follows. Section 2 discusses related studies. Section 3 elaborates a background work which covers the broadcast disk architecture and empty slot issue. Section 4 presents our proposed work, which is an enhancement of broadcast Disk method. Section 5 presents the performance evaluation of our proposed scheme compared to the Broadcast Disk method and finally section 6 is the conclusion.

2. RELATED STUDIES

A lot of work has been reported on data broadcasting [5 – 7], [19 – 22]. The Datacycle project [8] at Bellcore developed an alternative architecture for a database machine that was periodically broadcasted through a high

bandwidth communication channel. The storage's server pumps data to an arbitrary large number of access managers.

The Broadcast Disk model [5 – 7] is a pure push based data delivery model which achieves scalability by repeatedly broadcasting data items of common interest to a large client population, so that client with the same request can be served simultaneously. Based on the broadcast disk scheme, different hierarchies of slots are formed. The highest level of hierarchy is filled with hot data/pages and is broadcasted more frequently whereas low hierarchy is filled by cold data/pages and is broadcasted less frequently.

The hybrid data dissemination schemes [9], [10], [12], [19] proposed two data transmission modes delivery scheme: the periodic broadcast or push mode and the on demand broadcast or pull mode. Servers interleave the pushed and pulled data items based on an ad-hoc bandwidth partition parameter.

The use of clients' feedback and bit vectors is also proposed in [15 - 17]. It has been suggested that each client maintains its own bit vector to provide the client access statistics.

On-line algorithms are also proposed [11], [20], [21] to make the broadcasting data more adaptive and dynamic to the client access pattern. This work focuses only on scheduling, assuming that the pattern of interest is already known to the server.

Other model [18] showed that the servers schedule all the data items in database into a broadcast program and ignore the incurred computation overheads.

3. BACKGROUND

The environment to support the Broadcast Disks system is shown in Fig. 1. Data is fetched from the server's disk and broadcasted (through transmitter to the broadcast channel) to the mobile clients. The server applies push-based approach where data is broadcasted in one way direction to the mobile clients. At the server, data is organized in hierarchy, based on its popularity. Each hierarchy is referred as a disk. The most popular data sits on the first disk and the least popular data sits on the last disk. A program generator then interleaves (multiplexes) data within these disks to generate a series of slots which will be broadcasted to the mobile clients.

Fig. 2 is an example of a detail elaboration of the broadcast disk system. Assume that a list of pages (data) has been partitioned into three disks, *K1, K2* and *K3*. Pages of disk 1 are to be broadcasted three times faster than disk 3 and twice of disk 2. With this assumption, we set $R1 = 3$, $R2 = 2$ and $R3 = 1$, where R_i is the relative broadcast frequency of disk i. Each disk i split into NC_i (the number of chunks in disk i) chunks by first calculating L as the LCM (Least Common Multiple) of the relative frequencies and then split into $NC_i = L/R_i$ chunks. For example, $L = 6$ ($LCM(3,2,1)$), so $NC_1 = 2$, $NC_2 = 3$ and $NC_3 = 6$. Finally we create the broadcast program by interleaving the chunks of each disk in the following manner, where C_{ij} denotes the j-th chunk in disk i.

```
01 for i:= 0 to max_chunks -1
02     for j:= 1 to num_disks
03   Broadcast chunk C_{j,(i mod num_chunks(j))};
04     endfor
05 endfor
```

The resulting broadcast program consists of 6 minor cycles, and has 24 slots as shown in figure 2. These 24 slots will be repeatedly broadcasted to the mobile clients.

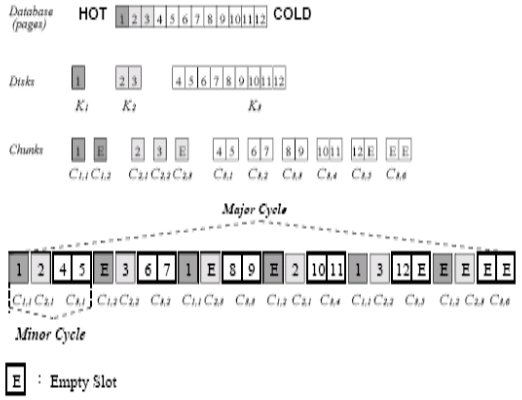


Fig. 2: Deriving broadcast program in the Broadcast Disk system.

3.1 Performance Issues

As mentioned in the previous section, a series of slots is generated and sent to the broadcast channel. The slots that not carrying data (empty), is considered as a wasted bandwidth. Fig. 2 shows an example of empty slots (represented by E). There are 8 empty slots out of 24 slots. It is observed that as the number of slots increases, the number of empty slots is also increases as shown in Fig. 3. Another issue with regards to empty slots is the clients' latency. Obviously, as the number of empty slots increases, the mobile clients must skip these empty slots before meeting the right slot and this increases their waiting time.

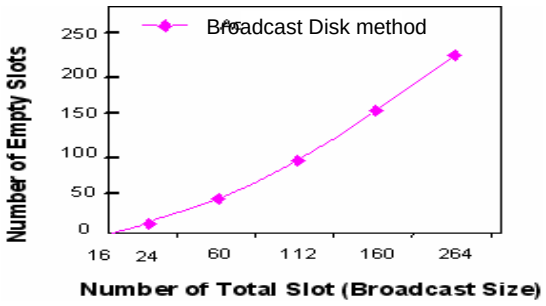


Fig. 3: Overall performance of empty slot in BD system.

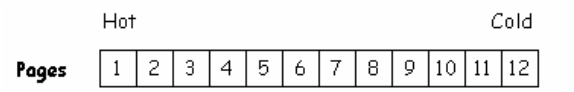
4. OPTIMUM BROADCAST DISK SYSTEM

In this section, we present a new broadcast disk system called the Optimum Broadcast Disk System. Our proposed work eliminates the empty slots issue faced by the existing broadcast disk system. The empty slots are automatically generated by the existing broadcast disk system and it is considered as a waste of resources. The empty slots are randomly generated, hence difficult to be located. Our approach is capable of identifying and collecting these empty slots, and fills them with data. As empty slots are filled with selected data (e.g. more popular data), the performance in terms of average users' waiting time is improved.

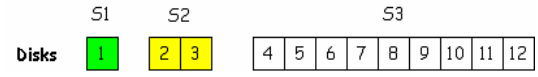
The detail algorithm of our method is elaborated below

and we use variables as shown in Table 1.

Step 1: Order the pages from the most popular to the least popular. E.g.: Given below is a list of twelve pages: page 1, page 2, up to page 12. Pages are arranged in decrement order from the most popular (hot) to the least popular (cold).



Step 2: Partition the pages into multiple ranges, where each range contains pages with similar access probabilities. These ranges are referred as disks S_i. For instance, the list is divided into three disks, S_1, S_2 and S_3.



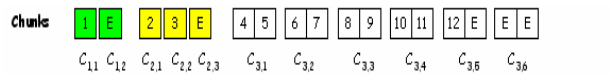
The example shows that disk S_1 contains the most popular page which is page 1. Disk S_2 contains less popular pages which are page 2 and page 3. Disk S_3 contains the least popular pages, which are page 4 to page 12.

Table 1: Variables used in proposed algorithm

Symbol	Variables
n	The number of disks
S_i	The i-th disk $1 < i < n$
S_i [a]	An array: the content of disk i at location a, a is integer
R_i	The relative frequency of disk i $1 \leq i \leq n$;
L	The least common multiple (LCM) of R_i, $1 \leq i \leq n$, i.e., $L=LCM$ (R_1, R_2,...R_s);
NC_i	The number of chunks in disk i, and $NC_i = L / R_i$, $1 \leq i \leq n$;
NS_i	The number of pages disk i, $1 < i < n$,
C_{ij}	The j-th chunk in disk i, $1 < i < n$;
C_i [b]	An array: the content of disk i at location b, b is integer
max_chunks	The maximum chunks of disks
num_chunks(i)	The number of chunks in disk i, $1 < i < n$
ES_i	Number of empty slots in disk i

Step 3, Set the relative frequency R_i for each disk. E.g.: Let's assume that there are three disks S_1, S_2 and S_3 and the pages in S_1 are broadcasted three times for every two times of S_2 and three times for every one time in S_3. Thus, the relative frequency of each disk is, $R_1 = 3$, $R_2 = 2$ and $R_3 = 1$.

Step 4, Split each disk into a number of units. These units are called chunks. Each disk is split into NC_i chunks by $NC_i = L/R_i$ where L is derived by having the LCM of frequency over R_i. E.g.: Having R_i in step 3 and $L = 6$ (LCM(3,2,1), we have $NC_1 = 2$, $NC_2 = 3$ and $NC_3 = 6$ with the following pattern:



The example shows that disk S_1 contains 2 chunks which are filled with page 1 and an empty chunk E. Disk S_2 contains of 3 chunks which are filled with page 2, page 3 and an empty chunk E. Disk S_3 contains of 12 chunks which are filled with page 4, 5, 6, 7, 8, 9, 10, 11, 12 and three empty chunks E.

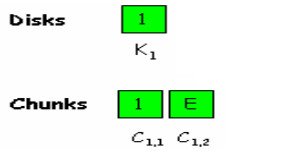
Step 5: Identify the empty chunks and fill them with pages by using the following pseudo code:

```
01  For i= 1 to n
02    If NC_i > n……..(i)
03      ES_i = NC_i – n.....(ii)
04     p = NC_i –ES_i……(iii)
05     For k=1 to ES_i
06       C [p + 1] = S[k]……(iv)
07       p = p + 1……(v)
```

The following is the elaboration of the pseudo code of step 5 using the output of 5 empty chunks as in step 4.

i. Identify which one of the NC_i contains empty chunks. This is done by checking if the number of chunks (NC_i) is larger than the number of chunks in S_i (if NC_i > n). If it is true, it means that the empty chunks present at NC_i of disk i.

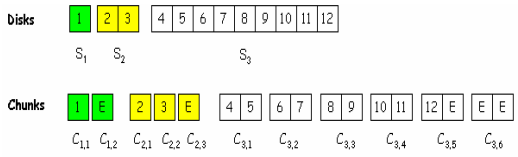
For example, let,



The number of chunks is greater than the number of disks (2 > 1). This concludes that the empty chunks exist.

ii. Find the total number of empty chunks in every NC_i (chunks of disk i). This is done by $ES_i = NC_i - n$.

For example, let,



- For disk S_1, we have NC_1=2 and S_1=1, hence the empty chunk ES_1 is 1 (2-1).
- For disk S_2, we have NC_2=3 and S_2=2, hence the empty chunk ES_2 is 1 (3-2).

- For disk S_3, we have NC_3=12 and S_3=9, hence the empty chunk ES_3 is 3 (12-9)

iii. Identify the exact location of the empty chunks at NC_i. This is done by:

Let,

The example shows that NC_1 = 2 and ES_1= 1, S_1 *[1]* = 1, C_1*[1]* = 1 and C_1*[2]* = E. By having $p = NC_i - ES_i$ then p = 1. Therefore the location of the empty chunk is $C[p + 1]$.

iv. Fill the empty slots with data. The empty slots of every NC_i, are filled with data by using:

For *k=1* to ES_1;

$C_1[p + 1] = S_1[k]$ # $C_1[2] = 1$ #

The empty chunk: $C_{1,2}$ now contains page 1

The final result is:

Finally, in **step 6**: we create the broadcast program by interleaving the chunks on each disk *i* using the following pseudo code:

```
01  for i:= 0 to max_chunks -1
02    for j:= 1 to n
03      Broadcast chunk C_{j,(i mod num_chunks(j))};
04    endfor
05  endfor
```

The three disks (S_1, S_2, and S_3) are interleaved in a single broadcast cycle which consists of 6 minor cycles, and has a period of 24 slots with no empty slots, as shown in Fig.4.

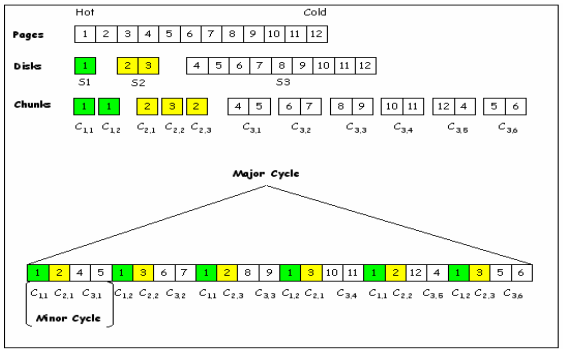


Fig. 4: A broadcast program with a new scheme without empty slots.

5. PERFORMANCE EVALUATION

This section presents the performance of our proposed scheme compared to the Broadcast Disk (BD) method. The evaluation is divided into two components: a case study which measures the empty slots' percentage of both schemes and a simulation which measures the average waiting time of both schemes.

5.1 Case Study

Table 2 shows the relative frequency R_i used for three broadcast disks indicated by R_1, R_2 and R_3. By using parameter

Table 2: Relative frequency for three broadcast disks

Broadcast	Δ	R_1	R_2	R_3
B1	1	3	2	1
B2	2	5	3	1
B3	3	7	4	1
B4	4	9	5	1

Case	Setting
B1	R_1=3, R_2=2, R_3=1 L=6(LCM(3,2,1) : NC_1=2, NC_2=3, NC_3=6
B2	R_1 = 5, R_2 = 3, R_3 = 1 L=15(LCM(5,3,1): NC_1=3, NC_2=5, NC_3 =15
B3	R_1 = 7, R_2 = 4, R_3 = 1 L=28(LCM(4,7,1), NC_1=7, NC_2=4, NC_3 =28
B4	R_1 = 9, R_2 = 5, R_3 = 1 L=45(LCM(9,5,1), NC_1=5 , NC_2 =9, NC_3=45

Δ, we can determine the relative frequency of each broadcast disk, R_1, R_2 and R_3 by computing as follows:

$$R_i = (S\text{-}i)\,\Delta + 1; \quad 1 \le i \le n.$$

Table 2 also shows four cases of B_1, B_2, B_3 and B_4 with their NC_i value and 12 pages are used for each case.

The results of the empty slots' percentage for both schemes are shown in Fig. 5. The percentage of empty slots in the Broadcast Disk method increases as the broadcast size increases. On the other hand, in our proposed method, as the broadcast size becomes larger, the percentage of the empty slots becomes zero.

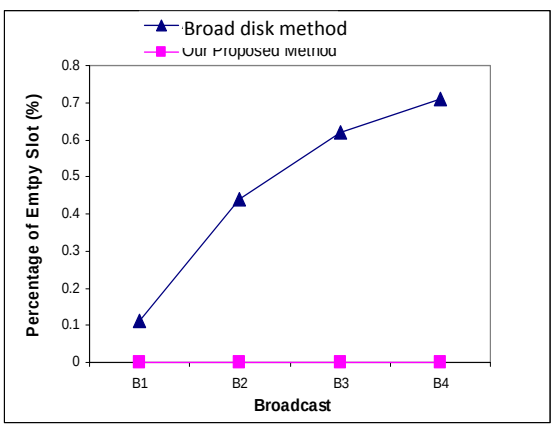


Fig. 5: The percentage of empty slots in four different broadcasts in the proposed method and BD method

5.2 Simulation

This section conducts a simulation to measure the clients' average waiting time in our proposed work compared to the Broadcast Disk method. We study the effect on the page popularity, users' arrival rate and number of pages. The popularity of pages is determined by Zipf distribution (z) [22] whereas the clients' arrival rate is determined by the Poisson distribution (p). The popularity of each page can be tuned by changing the Zipf distribution factor, z and the arrival rate by changing the Poisson distribution factor, p. We adopt the default value of z = 0.2 and p = 0.7 as they closely imitate the real world. The number of clients = 500 and the number of pages = 120. We use case $R_1 = 3$, $R_2 = 2$, $R_3 = 1$ which yield 240 slots. The development is done by using CSIM, on Pentium 128 MB running Window NT with Borland Compiler C++ version 5.

5.2.1 Average waiting time vs. Page popularity: The performance of the users' average waiting time against the page popularity is shown in Fig. 6. The x-axis represents z with different values which determine the difference in the popularity of each page. The popularity changes by having different values of z definitely affect the clients' average waiting time. Our scheme has reduced the average waiting time of the Broadcast Disk method by 40% regardless of the popularity changes. This is because those popular pages are copied into the empty slots. Fig. 6 shows us that as z become smaller, the average waiting time become lower too. This is because, as the z parameter becomes smaller, the access pattern becomes more localized with a small number of pages being picked by most of the clients.

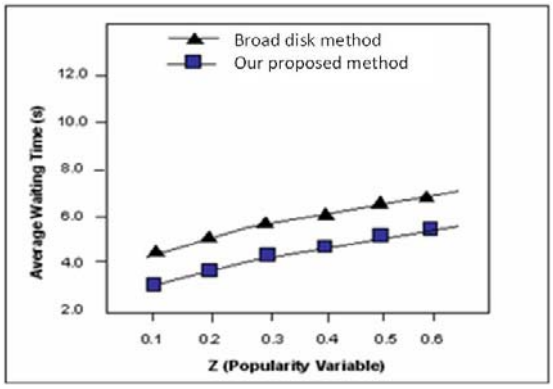


Fig. 6: The effect of page popularity on average waiting time (set $p = 0.7$)

5.2.2 Average waiting time vs. User arrival rate: The performance of users' average waiting time against users' arrival rate is shown in Fig. 7. The x-axis represents variable p with different values which determines the changes in the clients' arrival rate.

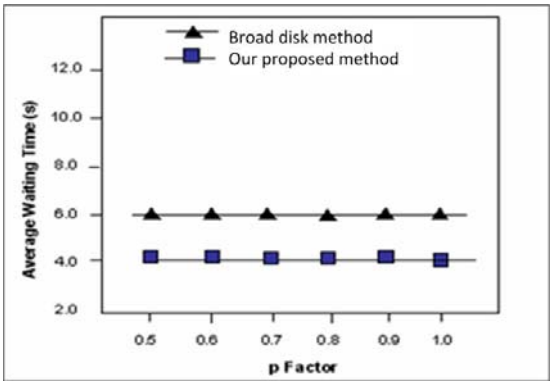


Fig. 7: The effect of users' access pattern on average waiting time (z = 0.2)

Fig. 7 shows that compared to the Broadcast Disk method, our method gives lower average waiting time regardless of the user arrival pattern. Our method gives 30% reduction in the average waiting time compared to the Broadcast Disk method.

5.2.3 Average waiting time vs. Pages : The performance of users' average waiting time against the number of pages is shown in fig. 8. The x-axis represents the number of pages. We set with p = 0.7 and z = 0.2 as it imitates the real world.

Fig. 8 shows that our scheme shows lower waiting time (as the number of pages increases), compared to the Broadcast Disk method. For example, the average waiting time for our method in the case of 10 pages is 2 seconds whereas in the Broadcast Disk method, it is 4 seconds. It shows that our proposed method has reduced 50% of the Broadcast Disk. As the number of pages increases, the percentage of users' average waiting time is increased as well. This is simply because of the increase in the number of empty slots due to the increase in the number of pages. As the number of empty slots increases and is filled with data, the users' average waiting time is tremendously decreased.

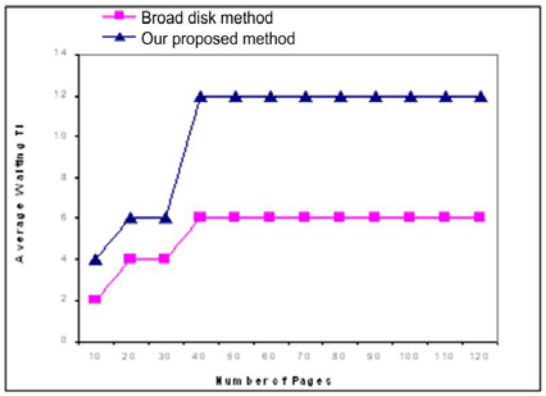


Fig. 8: The effect of pages on average waiting time

6. CONCLUSION

The rapidly expanding technology of cellular communications, wireless LAN, wireless data networks and satellite gives mobile users the ability of accessing information anywhere and anytime. One of the important

components of this environment is the broadcasting algorithm that resides in the server which is responsible of disseminating data to the mobile clients. One type of broadcasting method known as the Broadcast Disk creates a series of slots and broadcasts them in a circular manner on the air. Mobile clients keep listening to the air and catch those slots that interest them. The weaknesses of the Broadcast Disk method is the empty slot issue. Empty slots exist randomly in the series and difficult to be located. In this paper, we enhanced the Broadcast Disk method to overcome the empty slots issue and yet reduce the average waiting time. We managed to locate these empty slots and filled them up with the popular data. Our approach chose not to delete the empty slots instead copied the popular data into it. Our approach also did not alter the original structure of the Broadcast Disk system. We demonstrated through a case study and simulation to show that our scheme has eliminated the empty slots issue and has minimized the clients' average latency up to 30% regardless of the changes in page popularity and clients' arrival rate.

7. REFERENCES

[1] B.A. Myers, M. Beigl, "Handheld Computing", IEEE Computer Magazine, Vol. 36, No. 9, 2003, pp. 27 – 29.

[2] D. Barbara, "Mobile Computing and Database–A survey," IEEE Trans. Knowledge and Data Engineering, Vol. 11, 1999, pp. 108 – 117.

[3] J. Jing, A. Helal, and A.A. Elmagarmid, "Client-server Computing In Mobile Environments," ACM Computing Surveys, Vol. 31, No. 2, 1999, pp. 117 – 157.

[4] W. Wei, and V.R. Chinya, "Adaptive data broadcasting in Asymmetric Communication Environment", Proceeding of the International Database Engineering and Application Symposyium (IDEAS'04), 2004.

[5] Y.-I. Chang and C.–N. Yang, "A Complementary Approach to Data Broadcasting in Mobile Information System. A survey," IEEE Trans. Knowledge and Data Engineering, Vol. 40, No. 2, 2002, pp. 181 – 194.

[6] V. Liberatore, "Circular arrangement and cyclic broadcast scheduling", Journal of Algorithm, Vol. 51, No. 2, 2004, pp. 185 – 215.

[7] A. Acharya, M. Franklin and S. Zdonik, "Prefetching from a broadcast disk", Proc. of the International Conference on Data Engineering, New Orleans, LA, 1996, pp. 276 – 285.

[8] G. Herman et al., "The datacycle architecture for very large high throughput database systems," In Proceeding ACM SIGMOD Conference, San Francisco, 1997, pp. 97 – 103.

[9] Y.D. Chung, and M.H. Kim, "Effective data placement for wireless broadcast", Distributed and Parallel Databases, Vol. 9, No. 2, 2001.

[10] C. H. Hsu. G. Chen, A.L.P. Chen, "A Near Optimal Algorithm for Generating Broadcast program on multiple channels", Proceeding of the ACM International Conference on Information and Knowledge Management (CKIM), 2001, pp. 303 – 309.

[11] D. Katsaros, Y. Manolopoulos, "Broadcast program generation for webcasting", Data Knowledge and Engineering, Vol. 49, No. 1 2004, pp. 1 – 21.

[12] K. Stathatos, N. Rousopoulos and J.S. Baras, "Adaptive data broadcast in hybrid networks", in Proc. 23rd International Conference on Very Large Databases, September 1997.

[13] J. Juran, A.R. Hurson, N. Vijaykrishnan, and S. Kim, "Data organization and retrieval on parallel air channels:Performance and energy issues", ACM/Kluwer Wireless Networks, Vol. 10, No, 2, 2004, pp. 183 – 195.

[14] D.P. Agrawal, and Q.A. Zeng "Introduction to wir3eless and mobile systems", Florance Brooks/Cole (Thompson Leaarning centre Inc), 2003.

[15] G. Lee, S.C. Lo, and A.L.P. Chen "Data allocation on wireless broadcast channels for effieciemt querry processing", IEEE Transaction on Computer, Vol. 51 No. 10, 2002, pp. 1237 – 1252.

[16] N.H. Vaida and S. Hameed, "Scheduling data broadcast in asymmetric communication with nonuniform environments", Wireless Network, Vol. 5, No. 3, 1999, pp. 171 – 182.

[17] W.C. Peng, and M.S. Chen, "Efficient channel allocation tree generation for data broadcasting in a mobile computing environment' ACM/Kluwer Wireless Network, Vol. 9, No, 2, 2003, pp. 17 – 129.

[18] W.G. Yee, E. Omiecinski, and S.B. Navathe, "Efficient data broadcasting for broadcast disk arrays", (Tech. Rep. No. GIT-CC-02-20) Gergia Institute of Technology, Atlanta 2001.

[19] J.H. Hu, K. Yeung, G. Peng, and K. Leung" A Novel Push and pull hybrid data broadcast scheme for wireless information networks, In IEEE Int. Conf. on Communication, Vol. 3, 2000, pp. 1778 – 1782.

[20] A. B. Waluyo, B. Srinivasan, and D. Taniar, "Optimal broadcast channel for data dissemination in mobile database environment", In Proc. Of the APPT'03, LNCS, 2003, pp. 655 – 664.

[21] H.P. Hung, M.S. Chen, "On Exploring Channel allocation in the diverse data broadcasting environment IEEE ICDCS, 2005.

[22] H. Schwetman, CSIM18 - The simulation Engine in Mesquite Software, Inc, Austin, TX., 1996.

Received: June 2008. Accepted: Nov. 2009