データベースシステム 113－43
（１９９７．７．１５）

放送型情報提供システムにおけるキャッシュ管理方式

井上 潮，石川裕治，田辺雅則，箱守 聰

NTTデータ通信株式会社 情報科学研究所
E-mail: {uinoue, ishikawa, tanbe, hako}@lit.rd.nttdata.co.jp

本稿では，放送型とオンデマンド型の2種類の方式により情報提供が行われるモバイルコンピューティング環境において，サーバ側から周期的に放送されるデータをクライアント側でキャッシュするための管理方式について検討する。まず，データアクセスに要する待ち時間を短縮するためのキャッシュ管理方針について検討し，クライアントのデータアクセス確率，サーバのデータ放送頻度，及びデータ長に基づくアルゴリズムを提案する。次に，解析モデルを用いてアルゴリズムの性能を評価し，特にキャッシュのサイズが比較的小さい場合に，提案したアルゴリズムが優れていることを示す。

Caching Strategies for Wireless Data Broadcasts

Ushio Inoue, Yuji Ishikawa, Masanori Tanabe, and Satoshi Hakomori

Laboratory for Information Technology, NTT DATA Corporation
E-mail: {uinoue, ishikawa, tanbe, hako}@lit.rd.nttdata.co.jp

This paper studies caching strategies of a client for data items repeatedly broadcast by the server under a mobile computing environment, where data items are delivered by broadcast and on-demand modes. We study caching strategies for reducing the wait time required in data access and propose a new algorithm based on the access probability of the client, broadcast frequency of the server, and length of data items. Then we evaluate the performance of several algorithms using an analytical model and show that our algorithm is outstanding, especially when the cache size is rather small.

1 Introduction

The bandwidth of wireless communication networks in general is lower than that of fixed networks. When many clients access a server simultaneously under such a network environment, the communication channels become busy and the response time becomes very long. A solution to this problem is using a broadcast mode instead of on-demand mode for data delivery [Imie93]. In the broadcast mode, a server repeatedly broadcasts a data stream containing data items and each client gets a necessary data item when it arrives. The response time is independent of the number of clients, since the data stream is shared among clients and no request message is sent from the clients to the server. The data stream looks as if it were a virtual sequential access memory or disk on the air. The Broadcast disk proposed by Acharya et al.[Acha95] is based on this approach. Imielinski et al. also proposed a mechanism using both broadcast and on-demand modes from the viewpoint of energy efficiency [Imie94].

We are developing a new wireless information system, MobiCaster [Tana97], combining broadcast and on-demand modes of data delivery. Four technical issues are studied for improving the system performance. The first issue is what data should be broadcast. The server may have many data items to deliver to clients. If the server broadcasts all of the data items, the data stream becomes very long and clients will have to wait for a long time until their necessary data items arrive. On the other hand, if the server broadcasts only a few data items, most of the clients will access data items using the on-demand mode and the communication channels will be jammed. We proposed an algorithm based on an analytical model that minimizes the average response time of clients [Tana97]. Another issue at the server side is how often data should be broadcast. The server can broadcast different data items with different frequency. The frequent broadcast of data items popular among clients may reduce the wait time for such data items, but increase that for unpopular data items. Optimization of broadcast frequency of each data item is therefore necessary.

The third issue is what data should be cached at the client side. Caching broadcast data items dramatically reduces the wait time. However, since the capacity of client memory is usually limited, all of the broadcast data items cannot be kept in the cache. The selection of data items kept in the client cache is very important. The last issue is whether a client should wait or demand. Each client may get a data item by two ways. One is waiting until it is broadcast, the other is sending an on-demand request to the server. The wait time of the on-demand access depends on the traffic of the communication channel, which depends on how often clients use the on-demand mode. Each client should decide the access mode intelligently.

This paper is focused on the third issue on the client's side, caching strategies. We classify caching strategies from the viewpoint of how they reduce the wait time for broadcast data items, and we propose a new algorithm based on the access probability, broadcast frequency, and data length.

2 Related Work

The notion of repetitive broadcast of data was developed for the Datacycle architecture at Bellcore [Herm87]. Datacycle was designed for a database machine with special purpose hardware. More recently, Imielinski et al. at Rutgers University investigated repetitive broadcast under mobile computing environments [Imie93]. They also proposed a mechanism using both broadcast and on-demand [Imie94]. Their major concern was energy efficiency for limited battery life. Acharya et al. proposed the Broadcast disk with a control by broadcast frequency. They also studied the performance improvement obtained by using caching [Acha95]. However, caching strategies for broadcast data have not yet been sufficiently studied.

3 System Model

Figure 1 illustrates an information system combining broadcast and on-demand modes of data delivery. The system is composed of a server and clients. The server repetitively broadcasts a data stream containing data items, *D1, D2, ...Dn,* via a down-link channel. Each client listens to the data stream and receives data items when they arrive at the client side. Each client can also send on-demand request messages to the server via an up-link channel and get data items via another down-link channel.

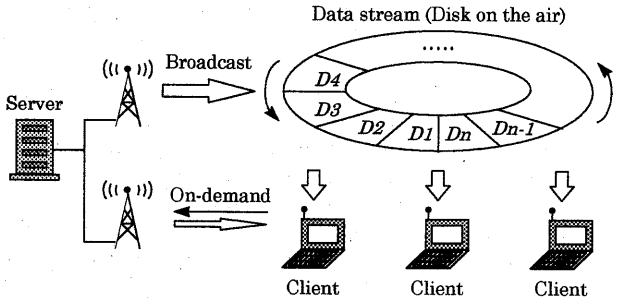

Figure 1: Information system with broadcast and on-demand modes

Figure 2 shows our system model of client software organization. Application programs (APs) demand necessary data items from the Data manager (DM). The DM checks the cache and if the data item is not kept in the cache the DM selects either of the access modes, broadcast or on-demand. If the on-demand mode is selected, the DM requests the Access manager (AM) to get the data item, and the AM sends a request message to the server via Up-link channel and gets the data item via Down-link channel 2. Otherwise, the DM waits until the Broadcast receiver (BR) receives the data item via Down-link channel 1. The DM also decides data items to be kept in the cache. When such data items are received by the BR, the DM stores them in the cache.

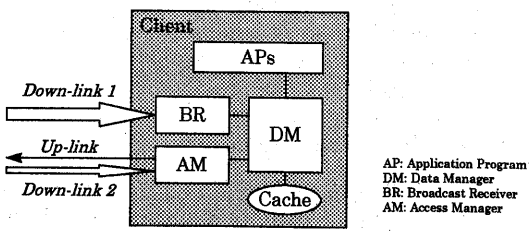

Figure 2: System model of client

We have two research goals on the technical issues in the DM. The first is to minimize the average wait time for demands of APs by keeping relevant broadcast data items in the cache (caching strategy). The second is to minimize the wait time by using the on-demand mode properly when the cache misses the necessary data item (demanding strategy). This paper discusses on caching strategies only.

4 Caching Strategies

4.1 Assumptions

We studied caching strategies for a kind of news delivery application, where many users receive hypertext data via wireless LAN with their palm top computers, which have no local disk storage. This paper assumes following conditions:

- The set of data items broadcast by the server is fixed. We ignore data items not broadcast by the server, because this paper is focused on caching strategies for broadcast data items.
- The broadcast frequency and position of each data item in the data stream is fixed. Clients know the configuration of the data stream by a timetable distributed by the server in advance.
- Each client knows its statistical access probability for each data item, but the DM (Data manager) does not know which data item will be demanded next by an AP (Application program).
- Although the broadcast frequency of each data item is reflected by the access probability of major clients, highly probable data items for each client may be different from others and not always be broadcast frequently.
- The capacity of the cache is far smaller than the total size of broadcast data items, and the wait time is negligible when the cache hits the data item.

4.2 Caching Strategies

The DM is required to manage the cache so that the average wait time of demands issued by APs is minimal. In other words, data items of higher cost to get from the network should be kept in the cache. For such cache management, the DM can utilize information on the access probability, broadcast frequency and position, and length of each data item.

The average wait time W, when the on-demand mode is not used, is calculated using the following formula:

$$W = \sum_i (Pi * Mi * Wi)$$

where Pi is the access probability, Mi is the

cache miss probability, and *Wi* is the expected wait time for the *i-th* data item. Two strategies exist: reducing *Mi* for large *Pi* or large *Wi*. The following caching algorithms are considered:

- RND (Random) algorithm selects any data items at random. It is the most simple, and reduces *Mi* uniformly.
- MAP (Most access probability) selects data items with high access probability. It reduces *Mi* for large *Pi*.
- LFB (Least frequently broadcast) selects data items with low broadcast frequency. It reduces *Mi* for large *Wi*.
- PIF (Probability inverse frequency) was proposed for the Broadcast disk [Acha95]. It selects data items with high *Pi/Fi*, where *Fi* is the broadcast frequency of the *i-th* data item. It is possible to reduce *Mi* for large *Pi* and *Wi*.
- PIFL (Probability inverse frequency & length) is our original algorithm, an improvement of PIF. It selects data items with high *Pi/Fi/Li*, where *Li* is the length of the *i-th* data item. PIFL is based on the observation that selecting shorter data items enables more data items to be kept in the cache and thus reduces *Mi* much more efficiently.

There are algorithms with different combinations, such as LDL (Least data length), PIL (Probability inverse length), IFL (Inverse frequency & length), and simple FIFO (First-in first-out), but the above five algorithms are sufficient to study performance improvement by caching.

5 Performance Evaluation

5.1 Parameters

We use a simple analytical model to evaluate the strategies described in the previous section. Table 1 shows the parameters and typical values to be used for our performance evaluation. These values were determined by supposing an application system that provides hypertext data written in HTML to diskless palm top computers via wireless LAN. The remainder of this subsection explains only the noticeable parameters.

For *Pi* and *Fi*, the 80-20% rule [Sacc82] is used. The 80-20% rule means 80% of access requests concentrate to 20% of the data items. In this paper, 20% of the data items accessed by clients with the higher probability are called hot data items, and 20% of the data items broadcast by the server with the higher frequency are called popular data items. As mentioned previously, hot and popular data items may be different because data items of interested to some clients are different from that of interest to other clients and not always broadcast frequently by the server. We define the similarity factor *S* between the two sets of hot and popular data items. In the case of *S=100%*, hot data items exactly match popular data items. In the case of *S=50%*, 50% of hot data items are popular but others are not popular. It also means 50% of popular data items are hot but others are not hot. In the case of *S=20%*, hot and popular data items are independent and have no mutual relation. It means 20% of hot data items are popular but others are not popular, and vice versa. Uniform distribution is assumed for each data item group throughout this paper so that our model becomes as simple as possible.

5.2 Broadcast Frequency

Assignment of broadcast frequency at the server is out of the scope of this paper, but has a impact on the performance of clients. This subsection, as a preliminary evaluation of the following subsections, studies the relation

Symbol	Parameter	Values
V	Bandwidth of Down-link channel 1	1,000 KB/s
N	Number of broadcast data items	1,000
Li	Length of i-th data item	25 or 100 KB (50% each)
Fi	Broadcast frequency of i-th data item	obeys 80-20% rule popular (*Fp*): 1-5, non-popular: 1
Pi	Access probability of i-th data item	obeys 80-20% rule hot: 80%, non-hot: 20%
S	Similarity factor between popular and hot data sets	100-20%
C	Cache memory size	variable

Table 1: Evaluation parameters and values

between the broadcast frequency of popular data items and the average wait time of clients without caching.

The cycle time of broadcast T, that is elapsed time to transfer a data stream, is calculated using the following formula:

$$T = \sum_j (Fj * Lj) / V$$

In the case where each copy of a popular data item is located uniformly in the data stream, the expected wait time Wi is as follows:

$$Wi = T / 2Fi$$

Thus the average wait time W is as follows:

$$W = \sum_i (Pi * Wi)$$

Figure 3 shows the relation between the broadcast frequency of popular data items Fp and the average wait time W, when the similarity factor is varied. In the case of $S=100\%$, W decreases as Fp increases, but W is almost constant when Fp is from 3 to 5. In the cases of $S=75\%$ and $S=50\%$, W is minimum at the point of Fp is 3 and 2, respectively. In the case of $S=20\%$, W increases as Fp increases. This is because T becomes longer as Fp increases, thus W goes higher when the similarity between hot and popular data items is low.

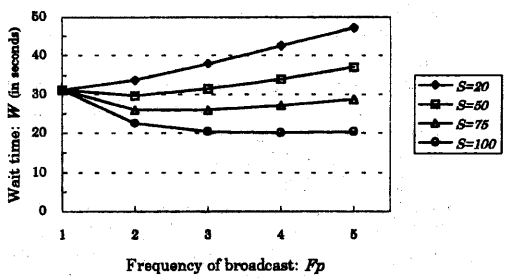

Figure 3: Frequency vs wait time w/o cache

According to the above observations, we fixed the value of Fp at 2 in the following evaluations.

5.3 Caching Strategies

This subsection evaluates the caching algorithms when the on-demand mode is not used. We study the relation between the cache size C and the average wait time W of clients. As mentioned previously, assuming that the wait time is negligible when the cache hits the data item, we calculate the average wait time using the following formula:

$$W = \sum_i (Pi * Mi * Wi)$$

$$Wi = T / 2Fi$$

- RND (Random) algorithm: The cache miss probability Mi is as following:

$$Mi = 1 - C / \sum_j Lj$$

- MAP (Most access probability):

$$Mi = \begin{cases} 0 & (Pi > Pt) \\ 1 & (Pi \le Pt) \end{cases}$$

where Pt is the threshold of the access probability. It is determined so that the following formula is satisfied:

$$C = \sum_i (Li * true(Pi > Pt))$$

where the $true(x)$ function returns 1 if x is true and returns 0 if x is false.

- LFB (Least frequently broadcast):

$$Mi = \begin{cases} 0 & (Fi \le Ft) \\ 1 & (Fi > Ft) \end{cases}$$

where Ft is the threshold of the broadcast frequency. It is determined so that the following formula is satisfied:

$$C = \sum_i (Li * true(Fi \le Ft))$$

- PIF (Probability inverse frequency):

$$Mi = \begin{cases} 0 & (Pi / Fi > Xt) \\ 1 & (Pi / Fi \le Xt) \end{cases}$$

where Xt is the threshold of the access probability divided by the broadcast frequency. It is determined so that the following formula is satisfied:

$$C = \sum_i (Li * true(Pi / Fi > Xt))$$

- PIFL (Probability inverse frequency & length):

$$Mi = \begin{cases} 0 & (Pi / (Fi * Li) > Yt) \\ 1 & (Pi / (Fi * Li) \le Yt) \end{cases}$$

where Yt is the threshold of the access probability divided by the broadcast frequency and data length. It is determined so that the following formula is satisfied:

$$C = \sum_i (Li * true(Pi / (Fi * Li) > Yt))$$

Figure 4 shows the relation between the cache size C normalized by the total size of data items and the average wait time W, in the case of $S=100\%$, for the five caching strategies. The performance improvement of RND and LFB is very little. W of MAP and PIF, both are identical in this case, decreases almost constantly as the cache size increases.

As we expected, W of PIFL is the smallest of all. The performance improvement is outstanding especially when C is from 4% to 8%.

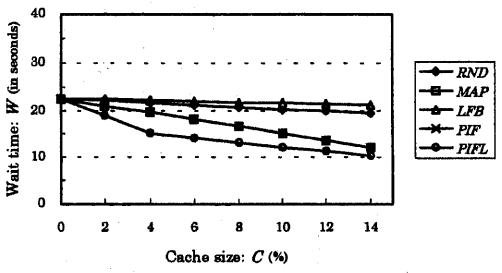

Figure 4: Cache size vs wait time when *S=100%*

Figure 5 is a similar graph for the case of *S=50%*. The performance improvement of MAP, PIF and PIFL looks better. This is because the wait time, when no cache is used, is longer than that in the case of *S=100%* due to the mismatch of hot and popular data items.

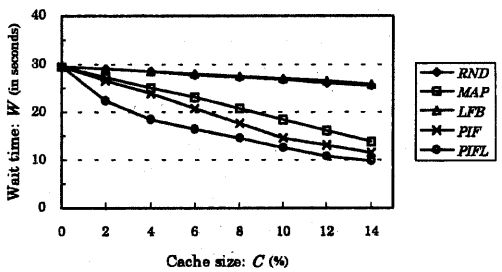

Figure 5: Cache size vs wait time when *S=50%*

6 Conclusion

This paper investigated caching strategies of a client for data broadcast by the server. We studied strategies from the viewpoint of how they reduce the wait time. Then we evaluated five caching algorithms by analytical models, including a new PIFL algorithm based on the access probability, broadcast frequency, and length of data items. The results show the performance improvement of PIFL is outstanding, especially when the cache size is rather small.

There are many other issues to be studied in future work. In this paper, we assumed the access probability for each data item is known; but in many cases, knowing such statistics in advance is difficult. New dynamic caching algorithms should be developed something like LRU (Least recently used) used under traditional on-demand only environments. We also assumed that the set of data items is fixed and data item values are not updated. Dynamic changes in the data require new efficient algorithms. As for the network environment, we assumed a single down-link channel was used for broadcasting; but in many wireless communication systems, multiple channels can be used. The server may broadcast a data item on different channels using different frequencies. This will have an impact on caching algorithms.

Regarding the performance evaluation, we used the uniform distribution based on the 80-20% rule throughout this paper. However, more general non-uniform distribution such as the Zipf [Gray94] should be used for more extensive evaluation. Moreover, the effects of interactions between broadcasting strategies at the server side and caching strategies at the client side should be evaluated.

References

[Acha95] Acharya, S., Alonso, R., Franklin, M., and Zdonik, S., "Broadcast Disks: Data Management for Asymmetric Communication Environments," Proc. of 1995 ACM SIGMOD Conf., pp. 199-210, June 1995.

[Gray94] Gray, J., Sundaresan, P., Englert, S., Baclawski, K., and Weinberger, P. J., "Quickly Generating Billion-Record Synthetic Databases," Proc. of 1994 ACM SIGMOD Conf., pp. 243-252, May 1994.

[Herm87] Herman, G., Gopal, G., Lee, K. C., and Weinrib, A., "The Datacycle Architecture for Very High Throughput Database Systems," Proc. of 1987 ACM SIGMOD Conf., pp. 97-103, May 1987.

[Imie93] Imielinski, T. and Badrinath, B. R., "Data Management for Mobile Computing," SIGMOD Record, Vol. 22, No. 1, pp. 34-39, Mar. 1993.

[Imie94] Imielinski, T. and Viswanathan, S., "Adaptive Wireless Information Systems," Tech. Report DCS-TR-312, Rutgers Univ., Oct. 1994.

[Sacc82] Sacco, G. M. and Schkolnick, M., "A Mechanism for Managing the Buffer Pool in a Relational Database System Using the Hot Set Model," Proc. of 8th Int. Conf. on Very Large Databases, pp. 257-262, Sep. 1982.

[Tana97] Tanabe, M., Hakomori, S., and Inoue, U., "Information-Providing Mechanism Combining Broadcast and On-Demand Modes in Mobile Environments," Proc. of 6th WINLAB Workshop, pp. 47-58, Mar. 1997.