# Time-Constrained Service on Air[*]

Yu-Chi Chung[1] Chao-Chun Chen[2] Chiang Lee[1]
[1]Department of Computer Science and Information Engineering
National Cheng-Kung University, Tainan, Taiwan, R.O.C.
{justim,leec}@dblab.csie.ncku.edu.tw
[2]Department of Information Management
Shih-Chien University Kaohsiung Campus, Taiwan, R.O.C.
chencc@mail3.kh.usc.edu.tw

## Abstract

*Data broadcasting is an efficient and highly scalable technique for delivering data to mobile clients in wireless environments. In this paper, we study the problem of scheduling broadcast data that are with an expected time within which the client is expecting to receive the data item. We analyze the problem and derive the minimum number of broadcast channels required for such a task. Also, we discuss the problems when the number of available channels is not enough. We propose novel solutions for both of the cases and the performance study indicates that our method is much better than the previous ones and performs very close to optimal.*

## 1 Introduction

With the fast advances of wireless technologies, mobile clients are able to conveniently acquire more and more information in real-time. In order to cope with the asymmetric communication property (that is, the downlink bandwidth is much higher than the uplink bandwidth) of such environments, *data broadcasting* becomes a natural solution for wireless communication applications. A client in such a system listens to the broadcast and waits until the required information comes and then downloads the data. The advantages of this broadcast type of data transmission are perfect scalability and high utilization of precious communication bandwidth [1].

Broadcast data are selected through a certain mechanism to ensure that they are needed by most clients, and then are scheduled in a particular order and broadcast in such an order, which is called a *broadcast program*. One of the major focuses of the research in the past on the scheduling of broadcast program is to reduce the *average access time*, that is, the time from a client's starting to listen (which could be anytime) until the required information is received [1, 2, 3, 5, 7, 10, 12, 15, 18]. However, the need of acquiring and even manipulating these broadcast data under certain constrained time is getting more and more demanding with the increasing involvement of services of mobile applications in our daily lives. For an example, the timing of buying/selling stocks for a stock holder is very crucial. If the stock information cannot reach a stock holder in time, the information might become useless. For another example, the information about traffic jam that is caused by a car accident should also reach a mobile client heading toward this direction timely. If a client receives such information early enough, the client is able to react accordingly to avoid the traffic jam. The value of the information would degrade significantly when the client gets closer to the spot of the accident. Another scenario is that a client may not be always patient in listening the broadcast information [9, 14]. When the waiting time is longer than the *expected time* of a client, the client could switch the access from a broadcast channel to an on-demand channel and actively sends a pull request through an uplink channel to ask for the desired data, instead of passively waiting in the broadcast channel for the data to come. Too often and too many such actions could seriously congest the on-demand channels. Therefore, the less can a broadcast program meet a client's expectation, the more serious congestion could be incurred in the on-demand channels [9]. If the waiting time of clients in the broadcast channels can be controlled within a certain expected time, the quality of service of the on-demand channels can also remain unaffected. These practical reasons and real-time requirements motivate the need of new research on the scheduling of broadcast data so as to fulfill clients' needs [8, 11, 14, 19].

In this paper, we propose scheduling mechanisms for

[*]This research was partially supported by the National Science Council in Taiwan under grant number NSC92-2213-E-006-043.

broadcasting data that are with an *expected time*. In this research, each data item (or formally a broadcast data page in our later presentation) is associated with an expected time. We assume that the importance of the data remains the same if a client receives the data within the expected time. Otherwise, its value diminishes or even becomes useless. Therefore, the broadcast program should be so designed that no matter when a client starts to listen, the client is always able to receive the required data either within the expected time or as close to the expected time as possible, if within it is impossible (due to lack of enough broadcast channels).

As for how to obtain the expected time of a data item, several previous research results can be applied to serve this purpose. The piggyback and the probing techniques are a few of those suitable for this purpose [4, 9, 13, 14, 16, 17].

A difficulty of scheduling the broadcast data so as to meet a client's expected time is that the client could start to listen at any time. The system has to guarantee that the client, no matter when to start to listen, will be able to receive all the desired data pages within the expected time. Several issues need to be considered in this research. (1) What is the minimally required number of the channels by which broadcast data are allocated so that clients are always able to receive the required data within the expected time? (2) If theoretically there exists such a minimum number of channels, then how to design a scheduling algorithm that can indeed utilize this minimum number of channels to serve the job (i.e., to let all clients receive the required data within the expected time)? (3) If the available channels of a broadcast system are insufficient (i.e., less than the theoretical minimum number of channels), then how to schedule the broadcast data so as to minimize the delay incurred by insufficient channels?

We propose solutions for all these issues. We first derive the theoretical bound for the number of channels that a broadcast system must have in order to broadcast data within their expected time. Then, we propose an optimal algorithm that always utilizes this minimum number of channels in scheduling the broadcast data. Also, we study how to schedule the data when the number of channels is insufficient and propose a heuristic algorithm to serve the task. Our performance results show that the proposed scheduling algorithm for insufficient channels achieves a much better performance than the past major solutions, and is very close to the optimal result.

The rest of the paper is organized as follows. In Section 2, we formally define our problem and the assumptions. Then in Section 3 we derive the required minimum number of channels and propose an algorithm that works under this number of channels. Another method working for the case of insufficient channels is proposed in Section 4. A preliminary performance study is then presented in Section 5. Finally, we conclude the paper in Section 6.

## 2 Problem Definition and Assumptions

We formulate our problem and make appropriate assumptions to make it a resolvable problem as follows. Given a number of data pages to be broadcast in multiple broadcast channels. Each of the data pages is associated with a known expected time. Every access of a client is only one data page. As the expected times may be different from each other, the problem can become too complicated to resolve. To make it resolvable, we assume that the expected times can be rearranged to $h$ groups, within each group the expected times of the data pages being equal. $t_{i+1}$ is equal to $c \cdot t_i$ for $1 \leq i \leq h-1$, where $t_i$ is the expected time of data pages of group $i$. For instance, 5 data pages need to be broadcast and their expected times are 2, 3, 4, 6, and 9, respectively. To schedule these data pages of almost arbitrary expected times, the complexity of the problem would be very high. A simple rearrangement of the expected times may drastically reduce the complexity of the problem. We reassign their expected times to 2, 2, 4, 4, and 8. That is, page 2, page 4, and page 5's expected times, originally being 3, 6, and 9, are now changed to 2, 4, and 8, respectively. Note that their new expected times are smaller than or equal to the original ones (i.e., $2 < 3$, $4 < 6$, $8 < 9$), so they still satisfy the requirements. Also, the new expected time should be as close to the original one as possible, so that the system does not waste the resource (i.e., bandwidth) by broadcasting this data page too early. In this particular example, there are three groups of expected times, and $t_1 = 2$, $t_2 = 4$, $t_3 = 8$, and $c = 2$ because $t_2 = 2 \cdot t_1$ and $t_3 = 2 \cdot t_2$. The reduction of the complexity of the scheduling problem will be seen in later sections. Also note that this rearrangement of expected times is not uncommon in the research of real-time paradigm. It is quite an elegant assumption to simplify the problem in that research area.

With this rearrangement, the expected times are classified to $h$ groups of identical expected times. Let these groups be $G_1, G_2, \cdots, G_h$. Also let the $j$-th data page of the $i$-th group be denoted as $p_{i,j}$, and the number of channels that a broadcast system really supply as $\mathcal{N}^{real}$. Let $P_i$ be the number of data pages of group $G_i$. Then, our research problem can be restated as follows. *Given data pages $p_{i,j}$ to be broadcast in $\mathcal{N}^{real}$ channels and $p_{i,j}$ being associated with an expected time $t_i$, where $1 \leq i \leq h$, and $1 \leq j \leq P_i$, the goal is to design such a scheduling algorithm that the produced broadcast program can satisfy all clients' needs as much as possible, regardless of the starting time of listening*. Following this definition, we find that the problem should be divided to two cases depending on the number of available channels. Let the required minimum number of channels be $\mathcal{N}$. If $\mathcal{N} \leq \mathcal{N}^{real}$, the design goal is to satisfy all accesses (i.e., to let every client receive the required data page within the expected time). If $\mathcal{N} > \mathcal{N}^{real}$, then meeting the expected time of every data page is impossible. So the design goal is to reduce the delay as much as possible. The two cases are presented in the following two sections respectively.

## 3 Method for Sufficient Channels

By sufficient channels, we mean that the number of channels supplied by the system is greater than or equal to the minimum number of channels required for the task, i.e., $\mathcal{N} \leq \mathcal{N}^{real}$. We first derive a theoretical lower bound of $\mathcal{N}$. Then, we provide a scheduling algorithm that is able to accomplish the task by utilizing the minimum number of channels.

### 3.1 Minimum Number of Channels

We refer to a *valid broadcast program* as the one that all data pages can be received by clients through this program within their expected times, no matter when a client starts to listen. Two conditions can be inferred from this definition of a valid broadcast program:

1. $p_{i,j}$ is broadcast at least once between the time 1 and $t_i$, where "1" means the time that a broadcast program starts.
2. The time between the $k$-th and the $(k+1)$-th broadcast of $p_{i,j}$, where $k \geq 1$, has to be less than or equal to $t_i$.

The number of data pages and how often these pages are broadcast directly affect the required number of channels. Their relationship is given the following theorem.

**Theorem 3.1.** ***(Minimum Number of Channels)***
*Let $\mathcal{N}$ be the minimum number of channels required for a valid broadcast program. $t_i$ is the expected time of data in group $G_i$ and $P_i$ is the number of data pages of $G_i$. Let $t_{i+1} = c \cdot t_i$, where $c$ is a positive integer and $i = 1, 2, \cdots, h$. Then,*

$$\mathcal{N} \geq \sum_{i=1}^{h} \left\lceil \frac{P_i}{t_i} \right\rceil \tag{1}$$

For a simple example, assume that two groups of data pages are to be broadcast. The numbers of pages of the two groups are 2 and 3, and their expected times are 2 and 4, respectively. Then, the minimum number of channels required for completing the job is

$$\mathcal{N} \geq \left\lceil \frac{2}{2} + \frac{3}{4} \right\rceil = \lceil 1.75 \rceil = 2$$

Any number of channels less than this minimum number (e.g., 1 channel in this example) is impossible to serve the job (i.e., data pages cannot be broadcast without causing a delay). The proof of the theorem can be found in [6]

### 3.2 A Scheduling Algorithm for Sufficient Channels

Now that the minimum number of channels is obtained, our next task is to design an algorithm that always generates a *valid broadcast program* and at the same time utilizes only the minimum number of channels.

Two conditions have to be satisfied for a broadcast program to be valid, as mentioned above. The "at least" part and the "less than" part of the two conditions indicate that there is a minimum number of times that a page should be broadcast. Hence, considering a given set of empty channels, Condition (1) implies that a data page with a smaller expected time should be assigned to the channels first. This is because if the front channel slots are occupied by data of a greater expected time, then eventually we may find no vacancies in the front of the channels for data pages of a smaller expected time so as to satisfy Condition (1). That is to say, Condition (1) implies an assigning order of data pages to the channels. Following that, Condition (2) tells how data are assigned to the rest of the channel slots. That is, a data page after the first assignment is repeatedly assigned to a channel slot for every $t_i$ time slots, if $t_i$ is the expected time of the data. As this algorithm operates under sufficient channels, we call it the *Scheduling Under Sufficient Channels (SUSC)* algorithm. Conceptually, the algorithm is greedy-based.

Let $\mathcal{B}$ be such a multi-channel broadcast program designed by the SUSC algorithm. In effect, $\mathcal{B}$ can be thought of as a two-dimensional array type of data structure, where each row represents a broadcast channel and each column has a set of time slots containing data pages to be broadcast at the same time. Formally the SUSC algorithm is presented in the following.

1. Arrange all data pages to be broadcast in ascending order according to their expected time. Those data pages of the same expected time are from the same group $G_i$, and their order is unimportant.

2. Get the first page in the list obtained above, letting it be $p_{i,j}$.

3. Call the GetAvailableSlot subroutine. This subroutine searches in $\mathcal{B}$ an available (unused) time slot. The search starts from the first time slot of the first channel till the $t_i$-th time slot of the same channel. If an available time slot is found, then GetAvailableSlot returns this time slot. Otherwise, the search will go on to the next channel to find an available channel. Note that it is nontrivial that an available time slot can always be found in the designated window of channel slots. Theorem 3.2 shows that this time slot indeed always exists.

4. Let the returned time slot from the GetAvailableSlot subroutine be $(x, y)$, where $x$ is the channel number and $y$ the slot number. As $p_{i,j}$ should be broadcast $\lceil t_h/ti \rceil$ times, starting from $(x, y)$ we assign $p_{i,j}$ once for every $t_i$ time slots. That is, $p_{i,j}$ is assigned to slots $(x, y)$, $(x, y+t_i)$, $(x, y+2\cdot t_i)$, $\cdots$, $(x, y+(\lceil t_h/ti \rceil - 1)\cdot t_i)$.

5. Repeat steps 2 $\sim$ 4 until all data pages are assigned to $\mathcal{B}$.

Note that certain steps of the algorithm can be further optimized to reduce the execution time. For example, the search of an available slot in Step 3 need not be always starting from the first slot of every channel. As this further reduction is secondary to the context, we omit it in the presentation for ease of understanding the concept of the method. The details of the algorithm is given in Algorithm 1 and Algorithm 2.

**Algorithm 1:** Pseudo code of the SUSC Algorithm.

```
GIVEN: $\mathcal{N}$, $h$, $t_h$, and all data pages to be arranged in a
broadcast program $\mathcal{B}$.
FIND: A valid broadcast program.
Sorting all data pages in ascending order according to their
expected time ;
for $i = 1$ to $h$ do
    for $j = 1$ to $P_i$ do
        $(x, y)$ = GetAvailableSlot($\mathcal{N}$, $t_i$, $\mathcal{B}$) ;
        /*$\lceil t_h/t_i \rceil$ is the broadcast frequency of $p_{i,j}$ */
        for $k = 1$ to $\lceil t_h/t_i \rceil$ do $\mathcal{B}[x, y + (k-1)\cdot t_i] \leftarrow p_{i,j}$;
```

**Algorithm 2:** Pseudo code of GetAvailableSlot Algorithm.

```
GIVEN: $\mathcal{N}$, $t_i$, $\mathcal{B}$.
FIND: An available time slot to contain $p_{i,j}$.
for $x = 1$ to $\mathcal{N}$ do
    for $y = 1$ to $t_i$ do
        if $\mathcal{B}[x, y]$ is an available slot then return $(x, y)$ else return
        "Not found" ;
```

### 3.3 Validity of the SUSC Algorithm

Two main points need to be clarified in order to prove the validity of the algorithm. They are given in the following two theorems. Readers may refer to [6] for the proof of the theorems.

**Theorem 3.2.** ***There always exists an available time slot*** $(x, y)$ ***in*** $\mathcal{B}$***, where*** $1 \leq x \leq \mathcal{N}$ ***and*** $1 \leq y \leq t_i$***, in executing the GetAvailableSlot of the SUSC algorithm such that*** $p_{i,j}$ ***can be arranged in*** $(x, y)$***.***

**Theorem 3.3.** ***For data page*** $p_{i,j}$ ***whose first appearance in*** $\mathcal{B}$ ***is*** $(x, y)$***, its*** $k$***-th appearance must be at slot*** $(x, y + (k-1)t_i)$***, for all*** $1 \leq k \leq \lceil t_h/t_i \rceil$.

## 4 Scheduling Under Insufficient Channels

When the number of available channels is less than the minimum number given in Theorem 3.1, the SUSC algorithm becomes infeasible for data scheduling. New algorithms need to be designed for this situation.

We first consider two possible solutions.

- Simply drop some data pages to reduce the amount of data to be broadcast so that the expected time of all broadcast data can be satisfied.

- Do not drop any data pages. Instead, we reduce the number of times that a data page is broadcast so that all broadcast data can fit in the insufficiently provided channels.

The task for the first solution is to choose some data pages and drop them from the broadcast list. Once they are removed and the remaining data can fit in the provided channels, the assignment can then be accomplished by using the SUSC algorithms presented in the last section. Although this solution seems rather simple and effective, it is in fact inadequate because the problem is not completely resolved. Those clients who do not obtain data from the broadcast channels are forced to issue requests to the server and access data through the on-demand channels. This hence increases the client population of on-demand channels and as a result, the quality of service of the on-demand channels are still severely degraded by using this solution.

For these reasons, we take the second approach - reducing the number of times that a page is broadcast - as our solution to the problem. The tradeoff is that a client may not be able to receive the data within the expected time. Our idea is to equally disperse the delay caused by channel insufficiency to all broadcast data. However, because the expected times of these data are different, how to arrange them so that the delay of each data page remains about the same becomes a difficult task. Of course, if the number of available channels is far less than required,then no methods including ours can prevent clients from switching to the on-demand channels to obtain their data. But, only if the system-provided broadcast channels reach as few as 1/5 of the required minimum number of channels, the delay that a client experiences in using our method is quite limited and ignorable, and the performance becomes almost as good as the theoretically optimal performance. In this section, we propose this method, formally the *Progressively Approaching Minimum Average Delay (PAMAD)* method.

## 4.1 Model of Average Delay Time

Given a broadcast program, the average delay, denoted as $\mathcal{D}$, of all broadcast data is the sum of access probability times average delay of every data page. That is,

$$\mathcal{D} = \sum_{i=1}^{h}\sum_{j=1}^{P_i} prob_{access}(p_{i,j}) \cdot avg\_delay(p_{i,j})$$

where $prob_{access}(p_{i,j})$ is the access probability and $avg\_delay(p_{i,j})$ is the average delay of $p_{i,j}$. Suppose that each data page has the same chance of being accessed by clients. Then $prob_{access}(p_{i,j})$ should be $1/n$, where $n$ is the number of broadcast data pages.

The other term $avg\_delay(p_{i,j})$ in the above equation is obtained in this way. Given a data page $p_{i,j}$ in a broadcast program $\mathcal{B}$, the time interval between the $k$-th and the $(k+1)$-th appearances in $\mathcal{B}$ remains the same for all $k$ because our design rationale (that will be used in the design of the PAMAD algorithm) is that the delay should be equally dispersed among the broadcast data. Hence, this time interval, which is greater than $t_i$, can be considered to have two segments, a *delay period* and an *expecting period*. Figure 1 gives an illustration of these two periods, in which $t_k(p_{i,j})$ is the time of the $k$-th appearance of $p_{i,j}$. A client entering the broadcast system within the expecting period can receive the data without delay. On the other hand, a delay is inevitable if the client enters within the delay period.

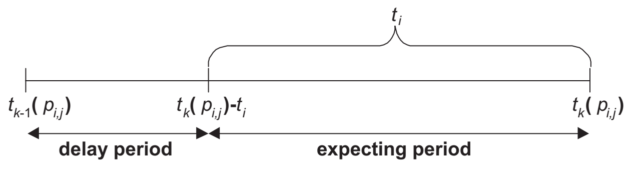


**Figure 1. An example of delay period and expecting period.**

As the probability of having a delayed access is proportional to the length of *delay period*, this probability can be represented as $\frac{(t_k(p_{i,j})-t_i)-t_{k-1}(p_{i,j})}{t_k(p_{i,j})-t_{k-1}(p_{i,j})}$. Upon the occurrence of delay (meaning that the client enters the broadcast system within the delay period), the average delay time should be one half of the delay period, i.e., $\frac{(t_k(p_{i,j})-t_i)-t_{k-1}(p_{i,j})}{2}$. Hence, the average delay caused by entering the broadcast system during the delay period is $\frac{(t_k(p_{i,j})-t_i)-t_{k-1}(p_{i,j})}{t_k(p_{i,j})-t_{k-1}(p_{i,j})} \cdot \frac{(t_k(p_{i,j})-t_i)-t_{k-1}(p_{i,j})}{2}$.

Let the number of times that $p_{i,j}$ is broadcast in $\mathcal{B}$ be $s_{i,j}$. The average delay caused by all broadcast data pages is therefore

$$\mathcal{D} = \sum_{i=1}^{h}\sum_{j=1}^{P_i} prob_{access}(p_{i,j}) \cdot avg\_delay(p_{i,j}) = \frac{1}{n}\cdot\sum_{i=1}^{h}\sum_{j=1}^{P_i}\frac{1}{s_{i,j}}$$
$$\left(\sum_{k=1}^{s_{i,j}} max\Big(\frac{(t_k(p_{i,j})-t_i)-t_{k-1}(p_{i,j})}{t_k(p_{i,j})-t_{k-1}(p_{i,j})}\cdot\frac{(t_k(p_{i,j})-t_i)-t_{k-1}(p_{i,j})}{2},0\Big)\right)$$

This expression indicates that the delay $\mathcal{D}$ is affected by two major factors, the time instant that a page is broadcast (i.e., $t_k(p_{i,j})$) and the number of times this page is broadcast (i.e., $s_{i,j}$). In the following, we first discuss how to determine $t_k(p_{i,j})$. Based on this result, we derive the optimal number of times that $p_{i,j}$ is broadcast which results in the smallest average delay.

## 4.2 Arrangement of $t_k(p_{i,j})$

As just stated, our design rationale is that delay should be equally dispersed among the broadcast data. Let $t_{major}$ be the time interval of a major cycle of $\mathcal{B}$. Then the time distance between this and the next appearances of $p_{i,j}$ in $\mathcal{B}$ should be $\lfloor t_{major}/s_{i,j}\rfloor$. Using this expression, we can rewrite $\mathcal{D}$ as follows.

$$\mathcal{D} = \frac{1}{n}\cdot\sum_{i=1}^{h}\sum_{j=1}^{P_i}\frac{1}{s_{i,j}}\left(s_{i,j}\cdot max\Big(\frac{(\lfloor\frac{t_{major}}{s_{i,j}}\rfloor - t_i)^2}{2\cdot\lfloor\frac{t_{major}}{s_{i,j}}\rfloor},0\Big)\right)$$

As $t_{major}$ and $t_i$ are both constant and known beforehand, $\mathcal{D}$ is therefore solely dependent on $s_{i,j}$, the number of times $p_{i,j}$ should appear in $\mathcal{B}$. This in turn determines the time instant that $p_{i,j}$ should appear in $\mathcal{B}$, i.e., $t_k(p_{i,j})$.

## 4.3 Derivation of Broadcast Frequency

The next problem is to determine the best $s_{i,j}$ such that $\mathcal{D}$ is minimum. Assume that there are $n$ pages to broadcast and let $s_{i,j}$ varies between 1 and $r$. Then, we have totally $r^n$ possible broadcast frequencies. Hence, a brute-force algorithm will lead to a solution of complexity $\mathcal{O}(r^n)$. Heuristics must be used to reduce the complexity of the problem.

As the expected time of data pages within a group is the same, their broadcast frequency should also be the same. Hence, we can reduce the problem from computing the broadcast frequency of each data page to that of each group. So the average delay obtained earlier becomes the average delay for a group (of data pages with the same expected time), or average group delay. Let $S_i$ be the number of times that group $G_i$ data are broadcast (i.e., the broadcast frequency of $G_i$). So $S_1\cdot P_1 + S_2\cdot P_2 + \ldots + S_h\cdot P_h$ is the number of all pages (containing multiple appearances of the same page) in $\mathcal{B}$. Let $\mathcal{F} = S_1\cdot P_1 + S_2\cdot P_2 + \ldots + S_h\cdot P_h$. Then, $prob_{access}(G_i)$ is equal to $\frac{S_i\cdot P_i}{\mathcal{F}}$. The average group delay, denoted as $\mathcal{D}'$, can be represented as follows.

$$\begin{aligned}\mathcal{D}' &= \sum_{i=1}^{h} prob_{access}(G_i)\cdot avg\_delay(G_i)\\ &= \sum_{i=1}^{h}\frac{S_i\cdot P_i}{\mathcal{F}}\cdot\Big(max\big((\frac{\mathcal{F}}{\mathcal{N}^{real}}\cdot\frac{1}{S_i}-t_i)\cdot(\frac{(\frac{t_{major}}{S_i})-t_i}{2}),0\big)\Big)\end{aligned} \quad (2)$$

In this expression, $\mathcal{N}^{real}$ is the number of channels really provided by the system (which is supposed to be less than $\mathcal{N}$ because we are considering the insufficient channel case now).

Our goal now is to find $S_1$, $S_2$, $\cdots$, $S_h$ for broadcasting data of $G_1$, $G_2$, $\cdots$, $G_h$ which result in the minimum average group delay. Our idea is this. Suppose that we already know the best frequencies $S_1$, $S_2$, $\cdots$, $S_{h-1}$ for broadcasting $G_1$, $G_2$, $\cdots$, $G_{h-1}$ when the length of a major cycle is $t_{h-1}$. Then, we may utilize this result in looking for the best $S_h$ for scheduling the next group $G_h$ in the broadcast program. This concept can in turn be applied to obtaining the best $S_{h-1}$ if the best combination of $S_1$, $\cdots$, $S_{h-2}$ is known. Following this, the whole problem can be divided into a number of subproblems and the result of each subproblem can be applied to resolving the subproblem of the next stage.

In addition to the above design rationale, a lower bound restriction is also employed in determining the broadcast frequencies. The restriction is that the data pages of every group have to be broadcast at least once in the program, even if some of the data pages of $G_i$ (for $1 \leq i \leq h$) may have to be arranged over the limit of $t_i$. This is to avoid the possibility that a data page may never be broadcast in the program. Our method proceeds in the following manner.

**Step 1:** The first step is to determine $S_1$ for broadcasting $G_1$ in $t_1$ time. As broadcasting once is enough for meeting the expected time $t_1$, $S_1$ is 1.

**Step 2:** Now we determine $S_1$ and $S_2$ for broadcasting $G_1$ and $G_2$, respectively, within $t_2$ which incurs the minimum average group delay. From this case on, we need to introduce a new parameter $r_i$ to represent the number of times that $G_i$ has been broadcast up to the $i$-th intermediate step. Note that $S_i$, different from $r_i$, is to represent the number of times that $G_i$ is broadcast in the entire broadcast program (i.e., after all $h$ groups of data are scheduled). Therefore, $r_i \leq S_i$.

As the expected time of $G_2$ is $t_2$, $G_2$ need only be broadcast once within $t_2$ time interval for the same reason as in Step 1. So $r_2 = S_2 = 1$. The rest of the channel slots within $t_2$ time can all be used to broadcast $G_1$, which is $r_1$ times. That is, for every allocation of $G_2$ in the program, $G_1$ would be assigned in the program $r_1$ times. Therefore, $S_1 = r_1 \cdot r_2 = r_1 \cdot 1 = r_1$ up to this stage. Substituting these for $S_2$ and $S_1$ in Equation 2, we have the average group delay of broadcasting the first two groups of data $G_1$ and $G_2$ (denoted as $\mathcal{D}'_2$) within a $t_2$ time interval.

$$\begin{aligned}\mathcal{D}'_2 = {} & \frac{r_1 \cdot P_1}{r_1 \cdot P_1 + P_2} \cdot max\left(\left(\frac{r_1 \cdot P_1 + P_2}{\mathcal{N}^{real}} \cdot \frac{1}{r_1} - t_1\right) \cdot \frac{\frac{t_{major}}{r_1} - t_1}{2}, 0\right) \\ & + \frac{P_2}{r_1 \cdot P_1 + P_2} \cdot max\left(\left(\frac{r_1 \cdot P_1 + P_2}{\mathcal{N}^{real}} - t_2\right) \cdot \frac{\frac{t_{major}}{1} - t_2}{2}, 0\right)\end{aligned} \tag{3}$$

Note that in this expression, only $t_{major}$ and $r_1$ are unknown. However, because $t_{major}$ represents the length of a broadcast program in which $G_1$ is broadcast $r_1$ times and $G_2$ is broadcast once, $t_{major}$ can be obtained as follows.

$$t_{major} = \left\lceil \frac{r_1 \cdot P_1 + 1 \cdot P_2}{\mathcal{N}^{real}} \right\rceil \tag{4}$$

That is, $t_{major}$ is a function of $r_1$. This in turn implies that $r_1$ is the only known parameter in $\mathcal{D}'_2$. By varying the value of $r_1$, we are able to obtain the optimal $r_1$, denoted as $r_1^{opt}$, that incurs a minimum $\mathcal{D}'_2$.

**Step 3:** Similar to Step 2, we determine $S_1$, $S_2$, and $S_3$ for broadcasting $G_1$, $G_2$, and $G_3$ respectively within a $t_3$ time interval and this broadcast program incurs the minimum average group delay. We know that $S_3 = r_3 = 1$ so that $G_3$ in this step need only be broadcast once for the same reason that has been stated in the previous steps. Also from the above discussion, we know that $S_2 = r_2 \cdot r_3 = r_2$, and $S_1 = r_1 \cdot r_2 \cdot r_3 = r_1 \cdot r_2$. Then, the average group delay of broadcasting the first three groups of data (denoted as $\mathcal{D}'_3$) can be rewritten as follows.

$$\begin{aligned}\mathcal{D}'_3 = {} & \frac{r_1 \cdot r_2 \cdot P_1}{r_2 \cdot (r_1 \cdot P_1 + P_2) + P_3} \cdot \\ & max\left(\left(\frac{r_2 \cdot (r_1 \cdot P_1 + P_2) + P_3}{\mathcal{N}^{real}} \cdot \frac{1}{r_2 \cdot r_1} - t_1\right) \cdot \frac{\frac{t_{major}}{r_1 \cdot r_2} - t_1}{2}, 0\right) \\ & + \frac{r_2 \cdot P_2}{r_2 \cdot (r_1 \cdot P_1 + P_2) + P_3} \cdot \\ & max\left(\left(\frac{r_2 \cdot (r_1 \cdot P_1 + P_2) + P_3}{\mathcal{N}^{real}} \cdot \frac{1}{r_2} - t_2\right) \cdot \frac{\frac{t_{major}}{r_2} - t_2}{2}, 0\right) \\ & + \frac{P_3}{r_2 \cdot (r_1 \cdot P_1 + P_2) + P_3} \cdot \\ & max\left(\left(\frac{r_2 \cdot (r_1 \cdot P_1 + P_2) + P_3}{\mathcal{N}^{real}} - t_3\right) \cdot \frac{t_{major} - t_3}{2}, 0\right)\end{aligned} \tag{5}$$

Same as in Step 2, $t_{major}$ can be obtained as follows.

$$t_{major} = \left\lceil \frac{r_2 \cdot r_1 \cdot P_1 + r_2 \cdot P_2 + P_3}{\mathcal{N}^{real}} \right\rceil \tag{6}$$

In this equation, $r_1$ is obtained from the last step (i.e., $r_1^{opt}$). Hence, only $r_2$ is unknown. That means $r_2$ is the only unknown variable in $\mathcal{D}'_3$. By varying $r_2$, we will be able to find the minimum $\mathcal{D}'$ and locate $r_2^{opt}$. Then, $S_1$ and $S_2$ are obtained.

**Step $h$:** Generalizing the above expressions, we have the average group delay of Step $h$ as follows.

$$\begin{aligned}\mathcal{D}'_h = {} & \sum_{i=1}^{h-1} \frac{\prod_{l=i}^{h-1} r_l \cdot P_i}{\sum_{j=1}^{h-1} \left(\prod_{k=j}^{h-1} r_k P_j\right) + P_h} \cdot \\ & \max\left(\left(\frac{\sum_{j=1}^{h-1} \left(\prod_{k=j}^{h-1} r_k P_j\right) + P_h}{\mathcal{N}^{real} \cdot \prod_{m=i}^{h-1} r_m} - t_i\right) \cdot \frac{\frac{t_{major}}{\prod_{n=i}^{h-1} r_n} - t_i}{2}, 0\right) \\ & + \frac{P_h}{\sum_{j=1}^{h-1} \left(\prod_{k=j}^{h-1} r_k P_j\right) + P_h} \cdot \\ & \max\left(\left(\frac{\sum_{j=1}^{h-1} \left(\prod_{k=j}^{h-1} r_k p_j\right) + P_h}{\mathcal{N}^{real}} - t_h\right) \cdot \frac{t_{major} - t_h}{2}, 0\right)\end{aligned} \tag{7}$$

By varying $r_{h-1}$, we will be able to determine $\mathcal{D}'_h$ and locate $r^{opt}_{h-1}$. Accordingly, we have all $S_i$'s as follows.

$$S_i = \begin{cases} \prod_{j=i}^{h-1} r_j & \text{if } i = 1, 2, \ldots, h-1 \\ 1 & \text{if } i = h \end{cases}$$

Algorithm 3 gives the pseudo code of deriving this set of broadcast frequencies $S_1$, $S_2$, $\cdots$, $S_h$. As Step 1 (i.e., $i$=1) is always trivial, it is omitted from Algorithm 3. Up to now, we have obtained the optimal broadcast frequencies of all groups of data. Our next step is to schedule these data pages according to the obtained broadcast frequency, which is presented next.

**Algorithm 3:** PAMAD_Calculate_Frequency Algorithm to deriving the broadcast frequencies of broadcast data.

```
GIVEN $\mathcal{N}^{real}$, $h$, $t_h$.
FIND A set of broadcast frequencies.
for $i = 2$ to $h$ do
    for $j = 1$ to $\lceil \frac{\mathcal{N}^{real} \cdot t_i - P_i}{\sum_{j=1}^{i-2}(\prod_{k=j}^{i-2} r_k \cdot P_j) + P_{i-1}} \rceil$ do
        if $min(\mathcal{D}'_i)$ occurs at $r^{opt}_{i-1}$ then $r_{i-1} \leftarrow r^{opt}_{i-1}$;
for $i = 1$ to $h$ do
    if $i = h$ then $S_i = 1$;
    else $S_i = \prod_{j=i}^{h-1} r_j$;
```

## 4.4 Generating Broadcast Program

Now we come to the last stage of the proposed PAMAD method. Knowing the broadcast frequencies of all data groups, we can schedule these data pages in the available channels. The details of this scheduling task is given in Algorithm 4. Their concepts are explained in the following.

The length of a broadcast cycle, having been mentioned previously, can be obtained by using the obtained broadcast frequencies $S_1$, $S_2$, $\cdots$, $S_h$.

$$\left\lceil \frac{\sum_{i=1}^{h} S_i \cdot P_i}{\mathcal{N}^{real}} \right\rceil \tag{8}$$

Within this major cycle, each data page of $G_i$ should be placed $S_i$ times. These placements should be evenly spread over the time slots of the major broadcast cycle. That means the interval between two placements of the same data page is $\frac{t_{major}}{S_i}$. However, as this may not be an integer, it should be adjusted to indicate exactly a time slot to hold the data page. Therefore, we have the exact spot for the $k$-th placement of a data page in the major cycle: $\lceil \frac{t_{major}}{S_i} \cdot (k-1) \rceil + 1$.

It is possible when placing the other data pages of the same group or a different group, the time slot $\lceil \frac{t_{major}}{S_i} \cdot (k-1) \rceil + 1$ of the first channel has been occupied by a previously placed data page. When this occurs, we simply search from the first channel to the last (i.e., the $\mathcal{N}^{real}$-th channel) for an empty space to hold the data page. If all of them are full, then we start from the next column (i.e., the $(\lceil \frac{t_{major}}{S_i} \cdot (k-1) \rceil + 2)$-th column) and repeat the same search process for an empty space until we reach the $(\lceil \frac{t_{major}}{S_i} \cdot k \rceil)$-th column. Within these columns of spaces, an empty time slot can always be found because the length of a major cycle has been calculated to hold all broadcast data pages. In this way, data pages of a group are evenly spread according to their broadcast frequency over the broadcast program.

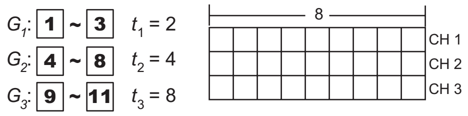
**Algorithm 4:** The Progressively Approaching Minimum Average Delay (PAMAD) Algorithm.

```
GIVEN $\mathcal{N}^{real}$, $h$, $t_h$, and all data pages to be arranged in a broadcast program.
FIND Broadcast program $\mathcal{B}$.
Call PAMAD_Calculate_Frequency (see Algorithm 3) to get the broadcast frequency of each group;
Sort all data pages in descending order according to their broadcast frequency;
$Required\_number\_of\_slots \leftarrow 0$;
for $i = 1$ to $h$ do
    $Required\_number\_of\_slots \leftarrow Required\_number\_of\_slots + S_i \cdot P_i$;
$t_{major} \leftarrow \lceil \frac{Required\_number\_of\_slots}{\mathcal{N}^{real}} \rceil$;
for $i = 1$ to $h$ do
    for $j = 1$ to $P_i$ do
        for $k = 1$ to $S_i$ do
            for $x = \lceil \frac{t_{major}}{S_i} \cdot (k-1) \rceil + 1$ to $\lceil \frac{t_{major}}{S_i} \cdot k \rceil$ do
                if $x > t_{major}$ then break ;
                for $y = 1$ to $\mathcal{N}^{real}$ do
                    if $\mathcal{B}[x, y]$ is an empty slot then $\mathcal{B}[x, y] \leftarrow p_{i,j}$;
```

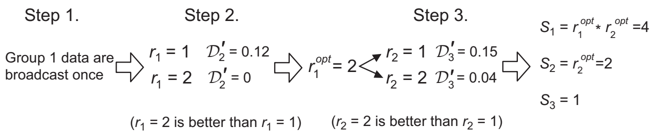
Let us use an example to illustrate how the PAMAD algorithm works. Figure 2(a) gives the data pages of three groups, $G_1$, $G_2$, and $G_3$, and their corresponding expected times, $t_1$, $t_2$, and $t_3$, respectively. $G_1$ has three data pages (page 1 to page 3), $G_2$ has five data pages (page 4 to page 8), and $G_3$ three data pages (page 9 to page 11). That is, $P_1 = 3$, $P_2 = 5$, and $P_3 = 3$. From Equation (1) we know that four channels are minimally required for broadcasting these groups of data pages without causing any delay. Assume that only three channels are available at this moment, as shown in Figure 2(a), so that the PAMAD algorithm is applied to design the broadcast program. In this method, the broadcast frequencies (referring to Section 4.3) of the groups of data are derived first. The process is given in Figure 2(b). **Step 1** is a trivial step in which $G_1$ data are broadcast just once within the $t_1$ time (where $t_1 = 2$). In **Step 2**, we compute how many times $G_1$ data should be broadcast within $t_2$ time (where $t_2 = 4$). From Equation (3), we find $\mathcal{D}'_2 = 0.12$ when $r_1 = 1$, and $\mathcal{D}'_2 = 0$ when $r_1 = 2$. As $G_1$ need only be broadcast at most twice within the first four time units (because at this step we are considering the broadcast frequency within time interval $t_2$) without incurring any delay, we do not have to consider the case of $r_1 \geq 3$. As $r_1 = 2$ gives the smallest $\mathcal{D}'_2$, we have $r^{opt}_1 = 2$.

In **Step 3**, we compute the number of times that $G_1$ and $G_2$ data should be broadcast within $t_3$ time (where $t_3 = 8$). We are going to obtain the $r_2^{opt}$ that causes the least $\mathcal{D}'_3$. From Equation (5) we find that $\mathcal{D}'_3 = 0.15$ at $r_1^{opt} = 2$ and $r_2 = 1$, and $\mathcal{D}'_3 = 0.04$ at $r_1^{opt} = 2$ and $r_2 = 2$. The cases for $r_2 \geq 3$ need not be considered further because broadcasting $G_2$ data twice is enough for the clients without causing any delay in retrieval. As $r_2 = 2$ results in a smaller $\mathcal{D}'_3$, we have $r_2^{opt} = 2$. Hence we have the broadcast frequencies for all three groups: $S_1 = r_1^{opt} \cdot r_2^{opt} = 4$, $S_2 = r_2^{opt} = 2$, and $S_3 = 1$, as given in Figure 2(b).

Next, the length of a major cycle can be obtained from Equation (8). In this example, the cycle length is $\lceil \frac{S_1 \cdot P_1 + S_2 \cdot P_2 + S_3 \cdot P_3}{3} \rceil = \lceil \frac{4 \cdot 3 + 2 \cdot 5 + 1 \cdot 3}{3} \rceil = \lceil \frac{25}{3} \rceil = 9$. Then, based on the derived optimal broadcast frequencies of the data groups, the data pages are evenly assigned and spread over the available channel space. Figure 2(c) gives an intermediate results after assigning $G_1$ data to the channels, and Figure 2(d) shows the finished broadcast program. Note that no matter when a client starts to listen to this final program, the client is guaranteed to experience on average a minimal delay in obtaining the required data.

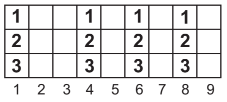


(a) Data to be broadcast.



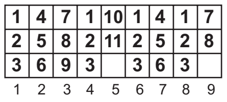
Step 1.
Step 2.
Step 3.
Group 1 data are broadcast once
$r_1 = 1$ $\mathcal{D}'_2 = 0.12$
$r_1 = 2$ $\mathcal{D}'_2 = 0$
$r_1^{opt} = 2$
$r_2 = 1$ $\mathcal{D}'_3 = 0.15$
$r_2 = 2$ $\mathcal{D}'_3 = 0.04$
$S_1 = r_1^{opt} * r_2^{opt} = 4$
$S_2 = r_2^{opt} = 2$
$S_3 = 1$
($r_1 = 2$ is better than $r_1 = 1$)
($r_2 = 2$ is better than $r_2 = 1$)

(b) Derivation of broadcast frequency.



| 1 |  |  | 1 |  | 1 |  | 1 |  |
|---|---|---|---|---|---|---|---|---|
| 2 |  |  | 2 |  | 2 |  | 2 |  |
| 3 |  |  | 3 |  | 3 |  | 3 |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

(c) The broadcast program after inserting $G_1$.



| 1 | 4 | 7 | 1 | 10 | 1 | 4 | 1 | 7 |
|---|---|---|---|---|---|---|---|---|
| 2 | 5 | 8 | 2 | 11 | 2 | 5 | 2 | 8 |
| 3 | 6 | 9 | 3 |  | 3 | 6 | 3 |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

(d) Final broadcast program.

**Figure 2. An illustrating example.**

## 5 Performance Evaluation

In the case of sufficient channels, we have presented an algorithm that always utilizes the minimum number of channels to broadcast. As all expected times are satisfied in this method, the problem has been resolved perfectly. Hence, nothing needs to be evaluated for this case. In the case of insufficient channels, however, the method needs to be carefully evaluated. We propose to use *average delay* (AvgD) as the performance metrics for evaluating the performance of a broadcast program under insufficient channels. Average delay is the time that on average a client has to wait in addition to the expected time for the desired data to come.

In order to investigate the performance under different data sets, the *broadcast data generator* creates data of $h$ groups, $G_1$, $G_2$, $\cdots$, $G_h$, and assign each group of data an expected time. Data of $G_1$ have the smallest expected time and those of $G_h$ have the greatest expected time. The number of data pages created for the groups follows one of the four group size distributions, i.e., *normal*, *S-skewed*, *L-skewed*, and *uniform* distributions. Figure 3 gives an illustration of the distributions. In all four distributions, the overall number of data pages are 1000.

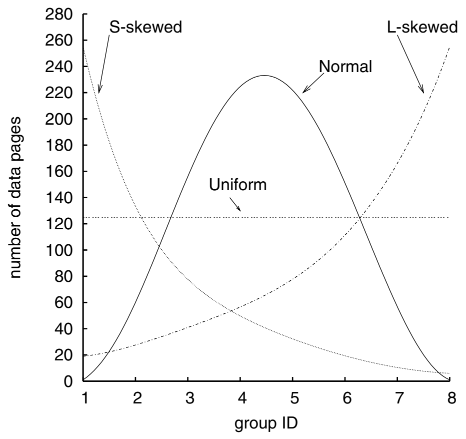


**Figure 3. Group size distributions of the groups.**

In order for a comparison, we choose a work in the literature that is closest to ours, the *periodic broadcast (PB) method* [19]. However, as the original paper of the PB method only provides an algorithm for a single channel environment [19], we have to extend their idea to cope with multiple channel environments. In order to make fair comparisons, assignment of data to multiple channels is the same as that of the PAMAD algorithm once the broadcast frequency is determined (referring to Algorithm 4 of the PAMAD method). We call this algorithm the modified PB (m-PB) algorithm.

Also for comparison purpose, we implemented an optimal (OPT) algorithm which exhaustively searches for a set of optimal broadcast frequencies that incurs the minimum delay. So this algorithm provides the optimal performance results, although its searching time is unacceptably high.

Figure 4 gives the default values used in the experiments. All the experiments are performed on a personal computer of Intel Pentium 2.4 GHz CPU with 1024 MB RAM running Windows 2000 operating systems.

| *Parameter* | *Default value* |
|---|---|
| $n$ - total number of data pages | 1000 |
| $h$ - number of groups (or expected times) | 8 |
| $t_i$ - expected time | 4, 8, 16, 32, 64, 128, 256, 512 |
| group size distributions | {normal,L-skewed,S-skewed,uniform} |
| number of requests | 3000 |

**Figure 4. Parameter settings.**

The evaluation result is shown in Figure 5. The number of channels varies from 1 to the minimum sufficient channels. In every subfigure, three algorithms, PAMAD, m-PB, and OPT, are compared. Three interesting facts are observed from these subfigures:

- In all four group size distributions, we found that the result of PAMAD almost overlaps with that of the OPT algorithm, and is much better than the m-PB method. This reveals that the PAMAD algorithm, although based on heuristics, achieves almost the optimal performance as we have expected in Section 4. And, the optimality of the algorithm is irrelevant to both the number of available channels and the type of group size distribution. That is, it performs extremely close to optimal regardless of the number of channels provided (server side factor) and the expected time (client side factor).

- This optimality also reveals that reducing broadcast frequency while channels are not enough is a much better approach than keeping the same broadcast frequency of a data page (which incurs a longer major broadcast cycle).

- From the figures we also found that when the number of available channels increases to about 1/5 of the minimally sufficient channels (i.e., the number of channels on the right end of the horizontal axis of the figures), the average delay AvgD of our method decreases to an amount almost ignorable. For one instance, the minimum sufficient channels is 64 in Figure 5(d). The value of AvgD declines dramatically when the number of channels increases from 1 to 10. After 10 channels, AvgD of our method becomes almost ignorable. This phenomenon appears in all four types of group size distributions. Hence, we conclude that in an environment where expected time constraints are not rigidly enforced or when the number of channels are not enough for running the SUSC algorithm, then as few as 1/5 of the minimally sufficient channels would be an ideal secondary choice, because with these channels almost the same good performance can be obtained.

## 6 Conclusions

In this paper, we studied the problem of time-constrained services on air. We proposed solutions for both the sufficient channel and the insufficient channel cases. When channels are more than the derived minimum number (i.e., sufficient channels), the designed SUSC algorithm provides optimal performance. When channels are insufficient, the proposed PAMAD algorithm still provides almost optimal performance by requiring as few as 1/5 of the minimum number of sufficient channels. Overall, the performance study reveals that the proposed solutions are very promising.

## References

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: data management for asymmetric communication environments. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 199–210, San Jose, CA USA, May 22-25 1995.

[2] S. Acharya, M. Franklin, and S. Zdonik. Dissemination-based data delivery using broadcast disks. *IEEE Personal Communications*, 2(6):50–60, December 1995.

[3] S. Acharya, M. Franklin, and S. Zdonik. Prefetching from a broadcast disk. In *Proceedings of the 12th International Conference on Data Engineering*, pages 276–285, New Orleans, Louisiana, February 26-March 1 1996.

[4] J. Cai and K.-L. Tan. Tuning integrated dissemination-based information systems. *Data & Knowledge Engineering*, 3(1):1–21, May 1999.

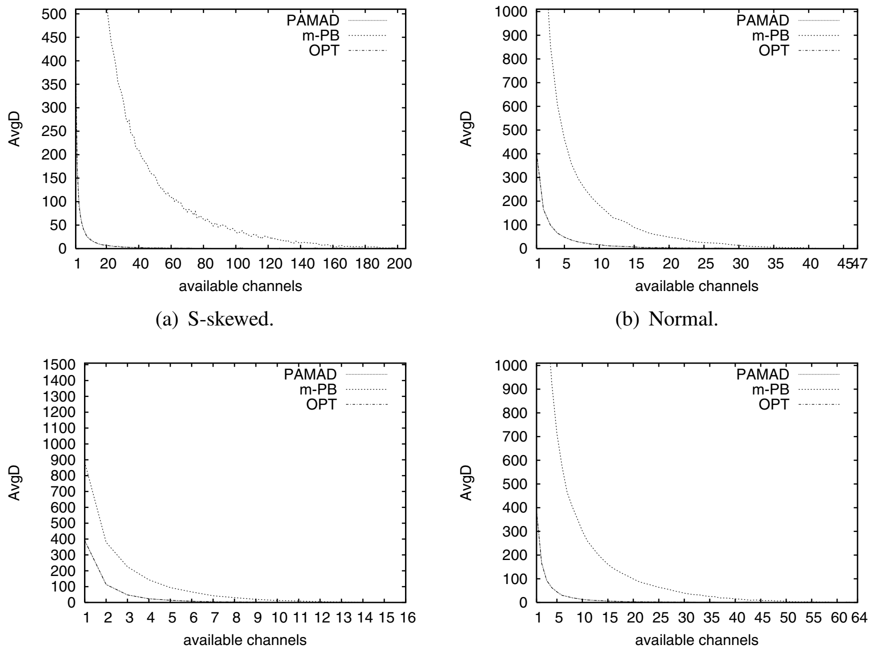
[5] C.-C. Chen, L.-F. Lin, and C. Lee. Benefit-oriented data retrieval in data broadcast environment. In *Proceedings of 9th International Conference on Database Systems for Advances Applications (DASFAA04)*, pages 916–921, Jeju Island, Korea, March 17-19 2004.

[6] Y.-C. Chung, C.-C. Chen, and C. Lee. Time-constrained service on air. Technical report, Department of Computer Science and Information Engineering National Cheng-Kung University, Tainan, Taiwan, R.O.C., http://dblab.csie.ncku.edu.tw/~justim/expected-time.ps, 2004.

[7] A. Datta, D. Vandermeer, A. Celik, and V. Kumar. Broadcast protocols to support efficient retrieval from database by mobile users. *ACM Transactions on Database Systems*, 1(24):1–79, March 1999.

[8] J. Fernandez-Conde and K. Ramamritham. Adaptive dissemination of data in time-critical asymmetric communication environments. In *Proceedings of the 11th Euromicro Conference on Real-Time Systems(ECRTS99)*, York, England, June 9-11 1999.

[9] C.-L. Hu and M.-S. Chen. Dynamic data broadcasting with traffic awareness. In *Proceeding of the 22nd International*



(a) S-skewed.

(b) Normal.

(c) L-skewed.

(d) Uniform.

**Figure 5. Average delay of four types of group size distributions.**

*Conference on Distributed Computing Systems (ICDCS'02)*, pages 112–119, Vienna, Austria, July 2-5 2002.

[10] Q. Hu, W.-C. Lee, and D. L. Lee. A hybrid index technique for power efficient data broadcast. *Distributed and Parallel Database*, 9(2):151–177, March 2001.

[11] K. A. Hua and S. Sheu. Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems. In *Proceedings of the ACM SIGCOMM'97 Conference*, pages 89–100, Cannes, France, September 1997.

[12] J.-H. Hwang, S. Cho, and C.-S. Hwang. Optimized scheduling on broadcast disks. In *Proceedings of the Second International Conference on Mobile Data Management (MDM01)*, pages 91–104, Hong Kong, Chian, January 8-10 2001.

[13] T. Imielinski and S. Viswanathan. Adaptive wireless information systems. In *Proceedings of the ACM Special Interest Group on DataBase Systems*, pages 19–41, 1994.

[14] S. Jiang and N. H. Vaidya. Scheduling data broadcast to "impatien" users. In *Proceedings of the ACM international workshop on Data engineering for wireless and mobile access*, pages 52–59, Seattle, WA USA, August 20 1999.

[15] K. Prabhakara, K. A. Hua, and J. Oh. Multi-level multi-channel air cache designs for broadcasting in a mobile environment. In *Proceesings of 13th International Conference on Data Engineering (ICDE 2000)*, pages 167–176, San Diego, CA, USA, February 28-March 3 2000.

[16] W.-C. L. Quinlong Hu, Dik Lun Lee. Dynamic data delivery in wireless communication environments. In *Proceedings of ER'98 Workshops on Mobile Data Access, volume 1552 of Lecture Nodes in Computer Science*, pages 218–229, 1998.

[17] K. Stathatos, N. Roussopoulos, and J. S. Baras. Adaptive data broadcast in hybrid networks. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 326–335, August 1997.

[18] P. Triantafillou, R. Harpantidou, and M. Paterakis. High performance data broadcasting: A comprehensive system's perspective. In *Proceedings of the Second International Conference on Mobile Data Management (MDM01)*, Hong Kong, Chian, January 8-10 2001.

[19] P. Xuan, S. Sen, O. Gonzalez, J. Fernandez, and K. Ramamritham. Broadcast on demand: Efficient and timely dissemination of data in mobile environments. In *Proceedings of the 3rd IEEE Real-Time Technology and Applications Symposium (RTAS '97)*, pages 38–48, Montreal, Canada, June 9-11 1997.