On The Efficient Use of Multiple Physical Channel Air Cache

Duc A. Tran Kien A. Hua
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32826, USA
Email: {dtran,kienhua}@cs.ucf.edu

Kiran Prabhakara
Oracle Corporation
Redwood Shores, CA 94065, USA
Email: Kiran.Prabhakara@oracle.com

***Abstract*— In limited and asymmetric bandwidth environments such as wireless networks, push strategy can be used to allow a large number of mobile users to access the shared data. Most of today's solutions assume that the server disseminates data on only one physical channel. In this paper, we focus on the problem of broadcasting data over multiple physical channels that cannot be coalesced into fewer high-bandwidth ones. This introduces new hurdles such as the heterogeneities of channels and data set. We propose a novel broadcast design at the server side, which is aimed at minimizing the response and download times. We also present a caching policy at the client side to further enhance the performance. Theoretical foundations and simulation-based studies are also presented to substantiate the near-optimality and to assess the advantages of the proposed technique.**

***Index Terms*— Data dissemination, broadcast, wireless networks, caching.**

I. Introduction

In recent years, mobile computing has attracted much attention due to its appealing computing environment. However the narrow bandwidth of wireless networks, and the relatively short active-life of power supplies (or batteries) of mobile units whose movement patterns are irregular make the problem of transmitting information a lot more challenging than in wired networks.

To overcome this obstacle, *push strategy* [1] has been used as an effective way of making the information available simultaneously to a large number of users. Rather than requiring users to explicitly request what they need as in the traditional *pull approach*, push-based techniques broadcast data in advance. Air is treated as a virtual cache and mobile units as users of that. This idea is very attractive and poses no bounds on the number of users reading from the air-cache. It also relieves the user of many burdens such as having to spend an inordinate amount of time polling known sites for updates and/or hunting on the network for relevant sites. Crucial to push-based approaches is the task of deciding what data to send and when to send them in the absence of specific requests. Researchers have proposed algorithms for designing broadcast schedules. These solutions use pure push as in *Broadcast Disks* [2], [3], [4], [5] and *hybrid* techniques (where most frequently accessed items are broadcast and the others are provided on demand) as in [6], [7], [8]. Other algorithms for scheduling broadcasts are introduced in [9], [10], [11], [12].

Those methods are based on the assumption that there exists a single physical channel for data broadcast. However, there are many scenarios where a server has access to multiple low-bandwidth physical channels, which cannot be combined to form a single high-bandwidth one. Some examples are provided below to justify this claim (more can be found in [13], [14], [15], [16]):

- Application Scalability: If the server application needs to be scaled to accommodate more mobile users, it may have to acquire additional physical channels. If these channels are in non-contiguous frequency ranges, they may have to be treated as separate channels when broadcasting data.
- Fault Tolerance: A base station can have more than one server with a transmission capability. Let us assume that servers A, B and C are broadcasting data in three non-contiguous frequency ranges, all in the same cell. Now, if servers B and C crash, the frequencies belonging to B and C need to be migrated to A in order to provide service to users of B and C. In such a case, these frequencies may not be combined to form one or more logical channels.
- Reconfiguration of adjoining cells: Let us assume there are two adjoining cells each with one server broadcasting at a different frequency range. If at some point in time, it is decided that a single server can serve both cells (by increasing the cell diameter), then the frequencies of one server needs to be migrated to the residual server. In this case also, the residual server gets multiple physical channels.

In this paper, we study the general case where the air cache consists of data items of various sizes broadcast on a number of channels with different bandwidths. The issues we address include how to partition the data set over the physical channels, and what ordering should be used to deliver a partition on its assigned channel. We also discuss in this paper how cache is managed at the client side to further improve the access time of a client.

The remainder of this paper is structured as follows. The details of the proposed technique are introduced in section II. Client designs are discussed in section III. Our performance study is reported in section IV. Finally in Section V, we give concluding remarks and provide pointers to future work.

II. Generalized Air Cache Design

A. Model

Our broadcast system consists of a server, which has access to a database and delivers data over known channels for mobile units to read the data. We assume a discrete model for time, bandwidth capacity and data size; a bandwidth of B means that we can send B data units in a time unit. The server channels may have different bandwidths. We denote by m the number of channels which are indexed as channel 1, channel 2, .., channel m. Each channel c ($1 \leq c \leq m$) has a bandwidth B_c and without loss of generality we assume that $B_1 \leq B_2 \leq .. \leq B_m$. The objects broadcast by the server are organized as "data items". In practice, these items could be web pages, stock data or database records that are requested by users. Data items can be organized as "self-identifying" [16] or "indexed" [17]. Broadcasting of indexed data items on multiple channels has been addressed in [14], [15]. Complementarily, our work focuses on "self-identifying" data items of various sizes. Let n be the number of data items and $s(i)$ the size of item i for each $i \in \{1, 2, .., n\}$. Each data item i associates with a weight $f(i)$ that reflects how often the item is accessed. They are normalized so that $\sum_{i=1}^{n} f(i) = 1$.

We use push-periodic as the delivery mechanism, and as in [6], we allow a supporting channel for mobile users to send requests up to the server. In response to a request, the server, based on the on-going schedule, informs the user of which channel to tune in for the requested data item. Since the supporting channel is only used for sending requests, not for receiving real data, the bandwidth consumed is infinitesimal. It is furthermore not necessary to broadcast any item on more than one channel, say channels *c* and *d*, because (1) doing this would increase the waiting time for the other items; and (2) the server always informs the client of the only channel (say channel c) where the requested item will be available earlier. In this case, there is no need to broadcast the item on channel *d*.

Our target clients are those having short-lived battery. Therefore, we are interested in minimizing the mean active time of a client session. Active time is computed as the delay between when the client sends a request and when the client finishes receiving the entire requested item. The *mean active time* (MAT) consists of

1) Mean Waiting Time (MWT): The average delay from when the client requests data to when the user starts receiving it.
2) Mean Download Time (MDT): The average time the client needs to download an item.

B. Server Design

In this subsection, we discuss broadcast techniques that are employed by the server. For convenience, we refer to an appearance of an item in the broadcast as an *instance* of the item. The *spacing* between two consecutive instances of an item is the distance in terms of data size between the beginning of the first instance and that of the second. The elapsed time of this spacing is computed based on the channel bandwidth. For example, from the current instance of an item if we have to send 100 Kbytes of data before the next instance, then the spacing between these two instances is 100Kbytes and a bandwidth of 50Kbytes/second would result in a 2-second elapsed time. For optimal broadcast scheduling on one channel it has been shown that all instances of an item should have the same spacing [11].

We denote a schedule S as ($[S_1, S_2, .., S_m]$, SP) where $SP : N \rightarrow R$ is the spacing mapping and each $S_i = \{i_1, i_2, .., i_{k_i}\}$ is the set of items broadcast on channel i ($\sum_{i=1}^{m} k_i = n$). Each instance of item i_j is $SP(i_j)$ apart from the next. Note that when we mention an item k_l, we implicitly know that this item is broadcast on channel k. Because a mobile unit is equally likely to request at any instant of time, the average time it has to wait until receiving the earliest instance of item i_j on channel i is $SP(i_j)/(2 \times B_i)$. Thus, we obtain the mean waiting time of a schedule S as MWT(S) = $\sum_{i=1}^{m}(\sum_{j=1}^{k_i}(f(i_j) \times SP(i_j))/(2 \times B_i))$. Also, since it takes $s(i_j)/B_i$ time units to download item i_j from channel i, the average download time resulted by schedule S is MDT(S) = $\sum_{i=1}^{m}(\sum_{j=1}^{k_i} f(i_j) \times s(i_j)/B_i)$. Since MAT = MWT + MDT, our approach to minimizing MAT is to take the minimizations of MWT and MDT into account. The theorems below give a theoretical basis for the proposed scheme:

Theorem 2.1: [MWT-Minimization]

Let S be a schedule that minimizes the mean waiting time. Then the following conditions must be satisfied:

1) $SP(i_j) = \sqrt{s(i_j)/f(i_j)} \times \sum_{l=1}^{k_i} \sqrt{f(i_l) \times s(i_l)}$
2) For any i between 1 and m, $\sum_{j=1}^{k_i} \sqrt{f(i_j) \times s(i_j)} = \sum_{l=1}^{m}\sum_{j=1}^{k_l} \sqrt{f(l_j) \times s(l_j)} \times B_i \,/\, \sum_{l=1}^{m} B_l$

The theorem will be proved using the lemma below:

Lemma 2.1: Assume that $X_1, X_2, .., X_m \geq 0$ and their sum is a constant C. $C_1, C_2, .., C_m$ are m positive constants. Then $X_1^2/C_1 + X_2^2/C_2 + .. + X_m^2/C_m$ is minimized if and only if $X_1/C_1 = X_2/C_2 = .. = X_m/C_m$.

Proof: [Lemma 2.1]

$X_1^2/C_1 + X_2^2/C_2 + .. + X_m^2/C_m = X_1^2/C_1 + X_2^2/C_2 + .. + X_{m-1}^2/C_{m-1} + (C - X_1 - .. - X_{m-1})^2/C_m$. This is a function of $m-1$ variables and we let it be $g(X_1, X_2, .., X_{m-1})$. Since $\frac{\partial^2 g}{\partial X_i^2} = 2(1/C_1 + 1/C_m) > 0$, function g is minimized when $\frac{\partial g}{\partial X_i}$ = $2(X_i/C_i - (C - X_1 - .. - X_{m-1})/C_m) = 0$ $\forall$ i from 1 to $m-1 \Leftrightarrow X_i/C_i = X_m/C_m$ $\forall i$ from 1 to m-1. $\Leftrightarrow X_1/C_1 = X_2/C_2 = .. = X_m/C_m$. ■

Proof: [MWT-Minimization]

Statement 1: For each item i_j, $x_{ij} = s(i_j)/SP(i_j)$ is the fraction of channel i's bandwidth allocated to item i_j. Therefore we have $\sum_{j=1}^{k_i} x_{ij} = 1$. Let us denote $a_{ij} = f(i_j) \times s(i_j)/(2 \times B_i)$. Since S minimizes MWT, given the set of items $\{i_1, i_2, .., i_{k_i}\}$ broadcast on channel i, the spacing values of those items must be chosen so as to minimize $T_i = (\sum_{j=1}^{k_i}(\mathrm{f}(i_j) \times \mathrm{SP}(i_j))/(2 \times B_i))$ where T_i represents the partial average waiting time on channel i.

It is clear that $T_i = \sum_{j=1}^{k_i} a_{ij}/x_{ij} = \sum_{j=1}^{k_i-1} a_{ij}/x_{ij} + a_{ik_i}/(1 - x_{i1} - .. - x_{i(k_i-1)})$. Let this be a function $h_i(x_{i1}, x_{i2}, .. , x_{i(k_i-1)})$ that has $k_i - 1$ independent variables. Since h_i has to be minimized, we must have $\frac{\partial h_i}{\partial x_{ij}} = 0 \ \forall j < k_i$ which is equivalent to the following: $-a_{ij}/x_{ij}^2 + a_{ik_i}/x_{ik_i}^2 = 0 \ \forall \ j < k_i \Leftrightarrow \sqrt{a_{ij}}/x_{ij} = \sqrt{a_{il}}/x_{il} \ \forall \ j, l \leq k_i \Leftrightarrow x_{ij} = \sqrt{a_{ij}}/\sum_{l=1}^{k_i}\sqrt{a_{il}} \ \forall \ j \leq k_i \Leftrightarrow s(i_j)/SP(i_j) = \sqrt{f(i_j) \times s(i_j)/(2 \times B_i)} \ / \sum_{l=1}^{k_i}\sqrt{f(i_l) \times s(i_l)/(2 \times B_i)} \Leftrightarrow SP(i_j) = \sqrt{s(i_j)/f(i_j)} \times \sum_{l=1}^{k_i}\sqrt{f(i_l) \times s(i_l)}$

Statement 2: We have MWT(S) = $\sum_{i=1}^{m}(\sum_{j=1}^{k_i}(f(i_j) \times SP(i_j))/(2 \times B_i))$. Replace SP(.) with the result from statement 1, we can rewrite MWT(S) = $\sum_{i=1}^{m}(\sum_{j=1}^{k_i}(\sqrt{f(i_j) \times s(i_j)}))^2 \ /(2 \times B_i))$. Let us denote: $Y_i = \sum_{j=1}^{k_i}\sqrt{f(i_j) \times s(i_j)}$, and $A = \sum_{i=1}^{m}\sum_{j=1}^{k_i}\sqrt{f(i_j) \times s(i_j)}$. Then we have $Y_1 + Y_2 + .. + Y_m = A$. Clearly, A is a constant for any schedule because A is the sum of every square-rooted multiplication of each item's size and weight.

MWT(S) = $Y_1^2/(2 \times B_1) + Y_2^2/(2 \times B_2) + .. + Y_m^2/(2 \times B_m)$. Following Lemma 2.1, we have: $Y_1/B_1 = Y_2/B_2 = .. = Y_m/B_m = (Y_1 + Y_2 + ... + Y_m) \ / \ (B_1 + B_2 + .. + B_m) \Leftrightarrow Y_i = B_i \times A \ / \sum_{j=1}^{m} B_j \Leftrightarrow \sum_{j=1}^{k_i}\sqrt{f(i_j) \times s(i_j)} = \sum_{l=1}^{m}\sum_{j=1}^{k_l}\sqrt{f(l_j) \times s(l_j)} \times B_i \ / \sum_{l=1}^{m} B_l$ ■

Theorem 2.2: [MDT-Minimization] Given S a schedule that minimizes the mean download time, for every pair of items (i_j, k_l) where $i < k$ (i.e., item i_j is broadcast on channel i, k_l on channel k, and $B_i \leq B_k$), one of the following must be true: (1) $B_i = B_k$, or (2) $f(i_j) \times s(i_j) \leq f(k_l) \times s(k_l)$.

Proof: [MDT-Minimization] By way of contradiction suppose that both the above statements are false. That is, $f(i_j) \times s(i_j) > f(k_l) \times s(k_l)$ and $B_i < B_k$ ($B_i > B_k$ cannot happen since $i < k$). We consider a new schedule S' that is similar to S except that item i_j and item k_l are now broadcast on channel k and i, respectively. We have $\Delta = MDT(S')$ - $MDT(S) \Rightarrow \Delta = f(k_l) \times s(k_l)/B_i + f(i_j) \times s(i_j)/B_k - f(i_j) \times s(i_j)/B_i - f(k_l) \times s(k_l)/B_k \Rightarrow \Delta = (f(k_l) \times s(k_l) - f(i_j) \times s(i_j))(1/B_i - 1/B_k)$.

$B_i < B_k$ implies $1/B_i - 1/B_k > 0$. In association with $T(i_j) \times S(i_j) > T(k_l) \times S(k_l)$, we have $\Delta < 0$. Therefore schedule S' is strictly "better" than S conflicting with the assumption that S is the MDT-optimal solution. ■

Since MAT = MWT + MDT, we expect both MWT and MDT to be small in order to make MAT small. However, to minimize both of these measures is not a trivial task in terms of efficiency and time-complexity. On the one hand, the optimal solution to the MDT-minimization problem is to broadcast every item on the fastest channel B_m. This solution would have a very high MWT. On the other hand, the minimization problem of MWT must be intractable since it is computationally harder than the Knapsack problem [18]. Therefore, instead of finding a solution having the optimal MAT, we propose a heuristic, but efficient way that tries to approximate the minimum MAT value. Specifically, according to the MDT-minimization theorem, in order to have small MDT, we should broadcast item i on a channel faster than the channel on which item j is broadcast if $f(i) \times s(i) > f(j) \times s(j)$. Furthermore from the MWT-minimization theorem, in order to have small MWT, the items $\{i_1, i_2, .., i_{k_i}\}$ to be broadcast on a channel i should minimize the difference between $\sum_{j=1}^{k_i}\sqrt{f(i_j) \times s(i_j)}$ and $A \times B_i/(B_1 + B_2 + .. + B_m)$.

The algorithm to distribute items over multiple channels is as follows:

Algorithm 2.1: [Dispatcher]

Input: A list L of n items $\{1, 2, .., n\}$

1) Sort L in the non-increasing order of $f(i) \times s(i)$
2) Let $ChannelIndex = m$, $i = 1$, $Variable = 0$, $A = \sum_{i=1}^{n}\sqrt{f(i) \times s(i)}$
3) Allocate the i^{th} element of L to channel $ChannelIndex$.
4) Increase $Variable$ by $\sqrt{f(i) \times s(i)}$
5) IF $Variable < A \times B_{ChannelIndex}/(B_1 + B_2 + .. + B_m)$:
   - Increment i. IF $i > n$ Exit
   - Go back to step 3.
6) ELSE
   - Decrement $ChannelIndex$, increment i, $Variable = 0$.
   - IF $ChannelIndex > 0$ and $i \leq n$ Go back to step 3
   - ELSE Exit

As a result of this algorithm, we know which items are to be broadcast on any channel. Given the spacing of each item determined by statement 1 of the MWT-minimization theorem, we need to schedule for them appropriately. Such a schedule can be made as in [11] which relates the problem to the *packet fair queuing* algorithm [19]. A similar online scheduler is presented as follows:

Algorithm 2.2: [Scheduler]

Input: A list L of k items $\{1, 2, .., k\}$ to be broadcast on a channel, as the result of Algorithm 2.1.

1) Compute the optimal spacing SP(j) for each item j using statement 1 of MWT-minimization theorem: SP(j) = $\sqrt{s(j)/f(j)} \times \sum_{j=1}^{k}\sqrt{f(j) \times s(j)}$.
2) The current time is denoted by $CurrentTime$, time to broadcast the current and next instance of item j by $Time_j$ and $Time'_j$. Initially, $CurrentTime = 0$, $Time_j = 0$ and $Time'_j$ = SP(j).
3) Select and broadcast the item j in L such that $Time_j \leq CurrentTime$ and $Time'_j$ is minimum.
4) $Time_j = Time'_j$; $Time'_j = Time_j$ + SP(j)
5) Once item j is completely transmitted, increase $CurrentTime$ by s(j) and go back to Step 3.

III. Client Caching

Client caching can be applied in several situations. For instance, in order to maximize a client's performance, it is a good idea to exploit the local memory of the client machine (with

caching capability) to cache data [2]. The delay is zero for those items residing in the cache. In an environment where a proxy server is used on behalf of a group of mobile units to contact the server, the proxy server can also leverage its local storage to cache items from the air cache [20]. Any client in this group can obtain a requested item very fast if it is available in the cache. Although it is applicable to the proxy level, for the sake of simplicity we assume that client caching is done at the client machine.

In [2], the authors proposed that client cache not simply the hottest items, but cache those items for which the local probability of access is significantly greater than the item's frequency of broadcast. This policy (called PIX) is very suitable for broadcast disks systems in which less frequently accessed items are broadcast on slower "disks". In our new technique designed for the generalized multi-channel model (heterogeneous channels and heterogeneous data items), that feature does not necessarily hold since we have to take sizes of items into account. If the client accesses an item frequently, that item should be in the cache. Further more, if the average active time a client needs to spend downloading an item from the air cache is very long, the item should also be in the cache. From this observation, if there are two items i and j such that $\beta(i) \times A_i > \beta(j) \times A_j$, and if we can cache just one item, we would choose item i. (Here, $\beta(.)$ is the local access probability of the client, and A_i and A_j the average active time for item i and j, respectively, to arrive on the air cache without using cache)

Let A be the sum of $\sqrt{f(.) \times s(.)}$ of all items in the database and B_c the bandwidth of the channel item i is broadcast on. Then $A_i = SP(i)/(2 \times B_c) + s(i)/B_c$. From MWT-Minimization theorem, we have $SP(i)/(2 \times B_c) = \sqrt{s(i)/f(i)} \times A \times B_c$ / $(2 \times \sum_{l=1}^{m} B_l)/B_c = \sqrt{s(i)/f(i)} \times A$ / $(2 \times \sum_{l=1}^{m} B_l)$. It implies that $A_i = \sqrt{s(i)/f(i)} \times A$ / $(2 \times \sum_{l=1}^{m} B_l) + s(i)/B_c$. Similarly, for an item j broadcast on channel d, $A_j = \sqrt{s(j)/f(j)} \times A/(2 \times \sum_{l=1}^{m} B_l) + B_d$. Therefore, $\beta(i) \times A_i > \beta(j) \times A_j \Leftrightarrow \beta(i) \times (\sqrt{s(i)/f(i)} \times A$ / $(2 \times \sum_{l=1}^{m} B_l) + s(i)/B_c) > \beta(j) \times (\sqrt{s(j)/f(j)} \times A$ / $(2 \times \sum_{l=1}^{m} B_l) + B_d)$.

Let $CacheSize$ be the size of client local memory used for caching purposes. Let $ActualSize$ be the total size of the current cache residents, and B the total server bandwidth. Let $C = A/(2 \times B)$. Each item i in the cache associates with a value $V(i)$ which is set to $\beta(i) \times (\sqrt{s(i)/f(i)} \times C + s(i)/B_d)$ when the item is stored in the cache at the first time. Here, d is the index of the channel broadcasting item i. Our caching replacement policy is described in detail below.

Algorithm 3.1: [Cache Replacement] Suppose that we consider bringing an item X from channel c to the cache.

1) Run a defragmentation on the cache
2) If $ActualSize + s(X) \leq CacheSize$, then item X is stored in the cache. No current cache resident is removed.
3) Otherwise, find all cache-resident items i's such that $V(i) < \beta(X) \times (\sqrt{s(X)/f(X)} \times C + s(X)/B_c)$ and $ActualSize - s(i) + s(X) \leq CacheSize$
4) Among those, select the item i having smallest $V(i)$ and replace it with X in the cache memory.
5) Set $V(X) = \beta(X) \times (\sqrt{s(X)/f(X)} \times C + s(X)/B_c)$
6) $ActualSize = ActualSize - s(i) + s(X)$

The above replacement algorithm requires that both global weight and local access distribution be known. When the client requests the item via a back connection, the global weight of an item can also be sent back by the server together with the channel index. For the local access probability, since our broadcast model does not assume a priori knowledge of the $\beta(i)$, the best we can do is to statistically estimate $\beta(i)$ for each item i, based on a history of requests to this item by the client. Let us call $\beta'(i)$ the probability estimate for item i. In the beginning the cache memory is free. When the client requests and then obtains a data item i, the item is stored in the cache and has $\beta'(i)$ set to 0. When item i is requested again, its probability estimate is updated using the following formula: $\beta'_{new}(i) = \alpha \times \beta'_{old}(i) + (1 - \alpha) \times K/D(K, i)$ where $\alpha \in [0, 1]$ is a constant controlling the relative weight of recent and past history in our prediction, K is a positive constant representing how far backward the history is taken into account, $D(K, i)$ is the period backward to the K^{th} most recent request to item i. In other words, $D(K, i) = x$ if since x units of time ago the client has requested item i for exactly K times, $D(K, i) = \infty$ if up to the current time item i has not been requested for at least K times.

Our approximation implementation is similar to the cache replacement policy in Algorithm 3.1 $\beta'(.)$ values are now used instead of $\beta(.)$ values.

IV. Performance Study

In this section, we present simulation results for the proposed technique. Let us use MCB (*M*ulti-*C*hannel *B*roadcasting) to refer to this technique. We used MWT (Mean Waiting Time), MDT (Mean Download Time) and MAT (Mean Active Time) as the performance measures. In our simulation model, the database is a collection of by default 1000 data items of possibly different sizes. The sizes are generated randomly and can be as large as 100 Kbytes. To model the information retrieval pattern, data items are requested according to a Zipf-like distribution [21] with a default skew factor z = 0.7. In each an-hour long simulation run, our workload generator generated requests on behalf of mobile users using an exponential distribution with a mean inter-arrival request rate of λ (requests/second). This rate in our experiments varied from 50 to 500 requests/second, which is enough to show the reasonable comparisons among the data delivery schemes. The default value of λ is 300 requests/second. We assumed that the server had multiple (50 by default) broadcast channels having bandwidths varying from 32Kb/s to 100Kb/s. We investigated the effect of data access pattern, database size (number of items), client cache size, and number of channels. The results are reported in the following subsections.

A. No cache at client

In this study we wanted to compare non-caching MCB with the optimal schedule. Since it seems unlikely to find out the schedule having the minimum MAT, we decided to estimate how our MWT and MDT are close to the optimal MWT and the lower bound of MDT, respectively. If MCB has near-minimal MWT and near-lower-bound MDT, we can conclude that its MAT is close to the optimal MAT. We compute the lower bound, upper bound for download time and minimum mean waiting time as follows: (The lower bound and upper bound for MDT are obtained if we put all items on the fastest channel and slowest channel, respectively) $Min(MWT) = (\sum_{i=1}^{m} \sum_{j=1}^{k_i} (f(i_j) \times s(i_j)))^2/(2 \times \sum_{i=1}^{m} B_i)$, $Lower(MDT) = \sum_{i=1}^{n} s(i) / B_m$ and $Upper(MDT) = \sum_{i=1}^{n} s(i) / B_1$.

In all simulation runs under the change of skew factor, number of items and number of channels, MCB consistently keeps very close to the optimal performance (about 1.3 times higher). The mean-waiting-time gap is narrower as the server has more bandwidth (Fig. 2) or smaller database size (Fig. 3). Under the access pattern (Fig. 1), both techniques result in lower waiting time under very skew distribution. When the distribution is very uniform, MCB is approximately 1.1 times the optimal mean waiting time. The download time is not affected under all changes, however MCB is a little higher than the optimal mean download time but too far from the worst case (with default parameters, MCB download time is 20% higher than the lower bound and 1/3 as large as the upper).

B. Caching at client

In this study, we assume that the server has been broadcasting data and there is only one requesting client. The local request probability is also a Zipf-like distribution however with a skew factor independent of the skew factor of the global access pattern. We first assess our caching policy under changes of caching space, ranging from 2MB (4% of the database size) to 20MB (40%). A caching space of 2MB is possible for a mobile device while 20MB is possible for current proxy servers. Fig. 4(Left) shows that MAT decreases quite linearly as more memory is added to the cache.

Figures 4(Right) and 5 illustrate the difference among the performance of MCB with different levels of client caching. Specifically we chose cache sizes of 5MB, 10MB and 20MB. The results show that MCB with a larger cache outperforms MCB with smaller cache and this gap is more significantly obvious as the server has very few channels (i,e., smaller bandwidth). In the simulation with default parameters, the non-cache MCB has a mean active time of 74 seconds while 5MB-Cache MCB has 52 seconds (30% gain), 10MB-Cache MCB has 40 seconds (46% gain) and 20MB-Cache MCB has 25 seconds (70% gain). Note that 5MB, 10MB and 20MB are equivalent to 10%, 20% and 40% of the entire database. This exhibits a significant advatange of using client cache.

V. Conclusions

Previous data broadcast methods assume the model in which the server delivers information on only one physical channel. However, there are practical situations where the server has access to multiple physical channels that cannot be combined to form a single higher-bandwidth one. We have presented a novel broadcast technique designed for a generalized model with multiple heterogeneous channels and various data item sizes. We have also discussed an efficient way to leverage local storage of clients for caching purposes. The proposed policies approximate the minimum active time promisingly by trying to minimize waiting and download times. The near-optimality is supported by mathematical proofs given in the paper and performance advantages are assessed by an in-depth simulation. One assumption of the proposed scheme is that the weights of items are known a priori. In environments where these values may vary as time goes by, we can re-determine them after every constant period based on the history of accesses. In such a period, the new scheme can be used to re-arrange the broadcast schedule.

In the near future, we are trying to adjust the proposed technique so that it can work with real-time data and without a priori knowledge about item weights. Additionally, it would be useful to evaluate techniques for broadcasting on multiple channels without letting the client know the channel index to tune in. In this case, we need to find an efficient method for the client to search the channel to join for the requested item.

References

[1] M. Franklin and S. Zdonik, "Data in your face: Push technology in perspective," in *Proc. of ACM SIGMOD*, Seattle, USA, June 1998.

[2] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast disks: Data management for asymmetric communications environments," in *Proc. of 1995 ACM SIGMOD*, May 1995, pp. 199–210.

[3] S. Acharya, M. Franklin, and S. Zdonik, "Disseminating updates on broadcast disks," in *Proc. of VLDB*, September 1996, pp. 354–365.

[4] A. Bar-Noy, B. Patt-Shamir, and I. Ziper, "Broadcast disks with polynomial cost functions," in *Proc. of IEEE INFOCOM*, Jerusalem, Israel., 2000.

[5] S. Baruah and A. Bestavros, "Pinwheel scheduling for fault-tolerant broadcast disks in real-time database systems," in *Proc. of the 13th IEEE DATA ENGINEERING*, Birmingham, U.K., April 1997, pp. 543–551.

[6] S. Acharya, M. Franklin, and S. Zdonik, "Balancing push and pull for data broadcast," in *Proc. of 1997 ACM SIGMOD*, May 1997, pp. 183–194.

[7] J. H. Oh, K. A. Hua, and K. Prabhakara, "A new broadcasting technique for an adaptive hybrid data delivery in wireless mobile network environment," in *Proc. of IEEE International Conference on Performance, Computing and Communications*, USA, 2000.

[8] K. Stathatos, N. Roussopoulos, and J.S. Baras, "Adaptive data broadcast in hybrid networks," in *Proc. of the 23rd VLDB Conf.*, Athens, Greece, 1997, pp. 326–335.

[9] A. Datta, A. Celik, J. Kim, D. VanderMeer, and V. Kumar, "Adaptive broadcast protocols to support efficient and energy conserving retrieval from databases in mobile computing environments," in *Proc. of the 13th IEEE DATA ENGINEERING*, Birmingham, U.K., April 1997, pp. 124–133.

[10] A. Datta, A. Celik, and V. Kumar, "Broadcast protocols to support efficient retrieval from databases by mobile users," *ACM Transactions on Database Systems*, vol. 24, no. 1, pp. 1–79, March 1999.

[11] S. Hameed and N. H. Vaidya, "Log-time algorithms for scheduling single and multiple channel data broadcast," in *Proc. of ACM MOBICOM*, Budapest, Hungary, 1997, pp. 90–99.

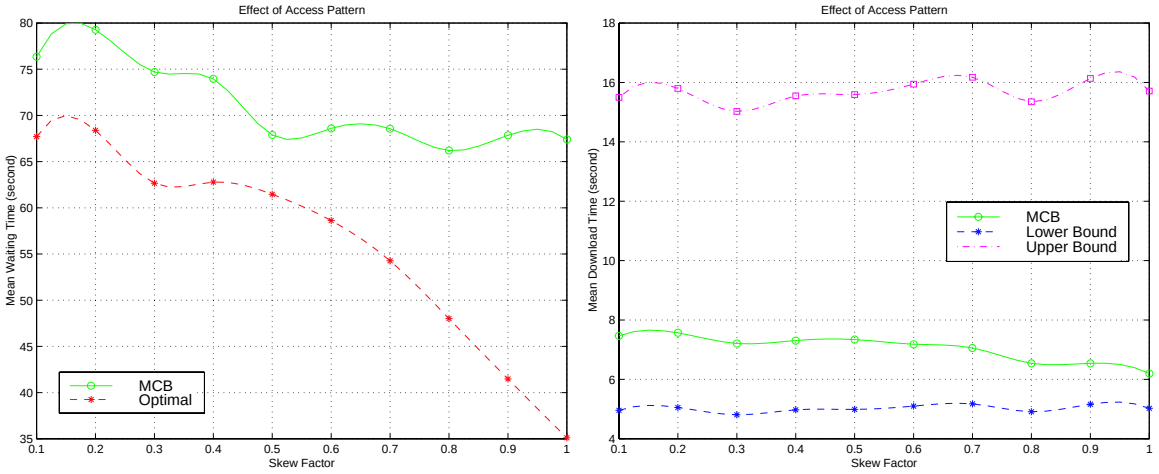

Fig. 1. Effect of global access pattern. Left: Mean Waiting Time; Right: Mean Download Time

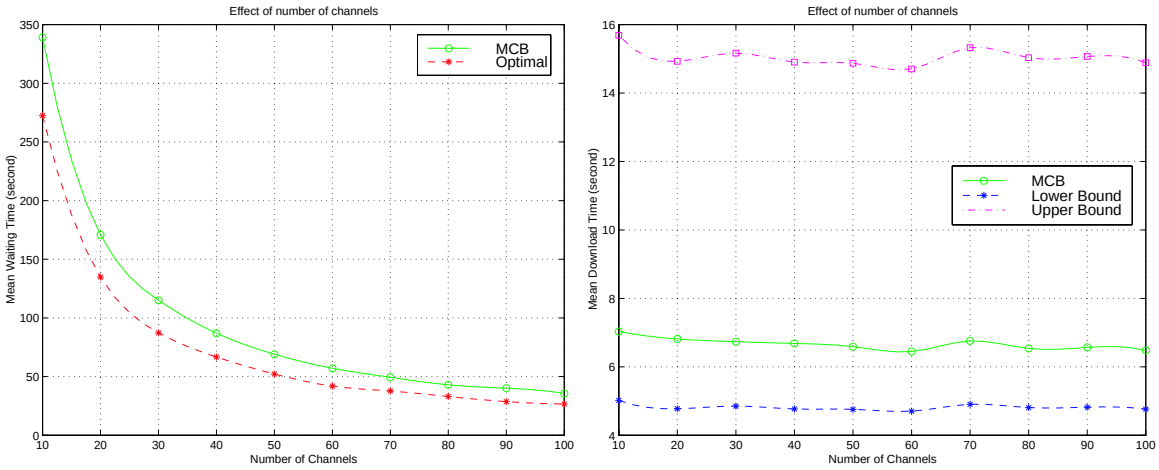

Fig. 2. Effect of number of channels. Left: Mean Waiting Time; Right: Mean Download Time

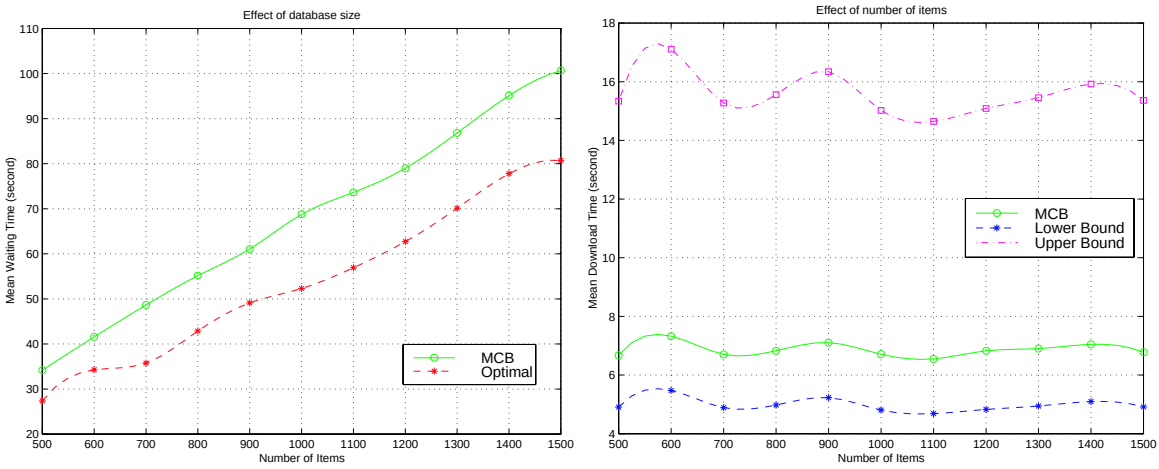

Fig. 3. Effect of database size. Left: Mean Waiting Time; Right: Mean Download Time

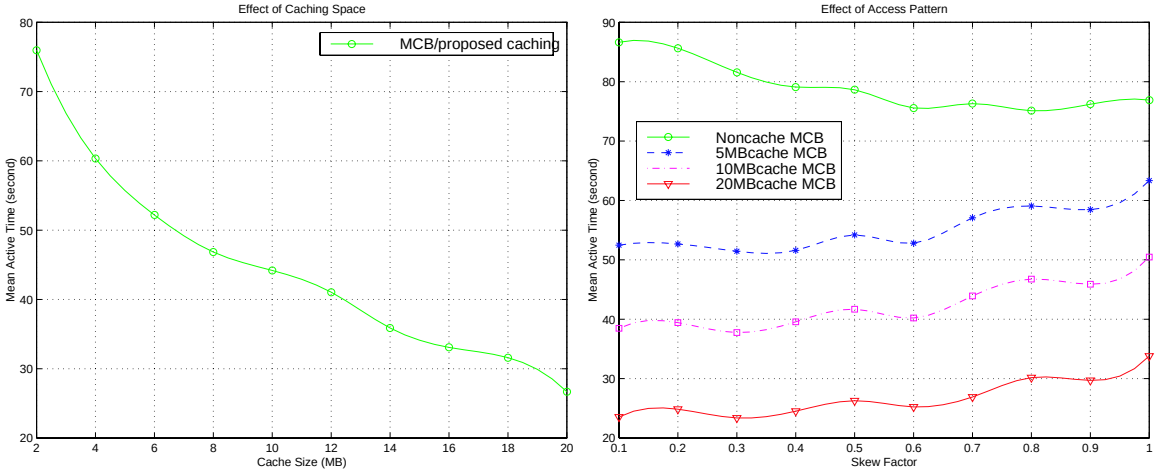

Fig. 4. Mean Active Time of MCB with caching; Left: Effect of Cache Size; Right: Effect of local access pattern

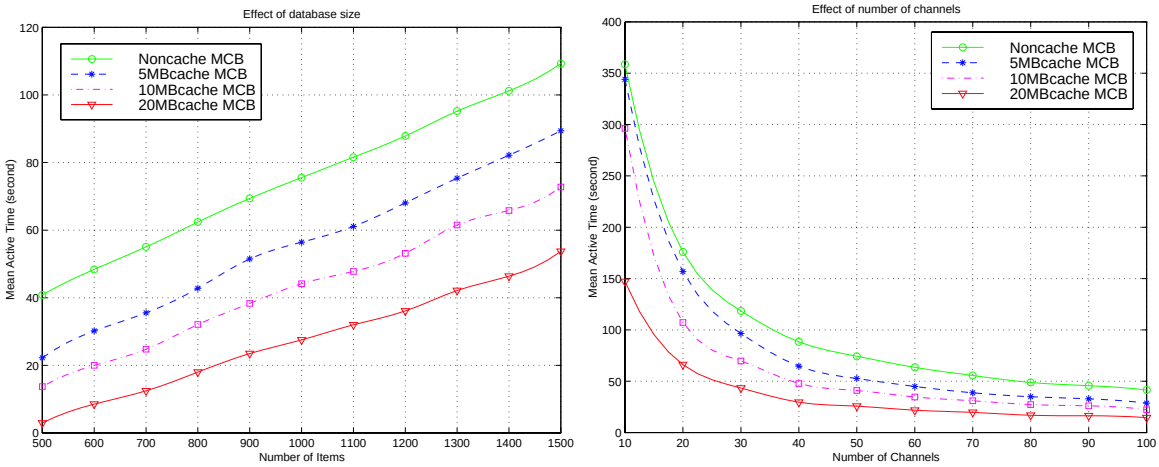

Fig. 5. Mean Active Time of MCB with caching; Left: Effect of database size; Right: Effect of number of channels

[12] S. Jiang and N. H. Vaidya, "Scheduling data broadcast to impatient users," in *Proc. of ACM MobiDE*, Seattle, USA, 1999, pp. 52–59.

[13] H. V. Leong and A. Si, "Data broadcasting strategies over multiple unreliable wireless channels," in *ACM CIKM '95*, Baltimore,MD, 1995.

[14] S.-C.Lo and A. L. P. Chen, "Optimal index and data allocation in multiple broadcast channels," in *IEEE Conference on Data Engineering*, 2000, pp. 293–302.

[15] N. Shivakumar and S. Venkatasubramanian, "Efficient indexing for broadcast based wireless systems," *ACM/Baltzer Mobile Network and Applications*, pp. 433–446, 1996.

[16] K. Prabhakara, K. A. Hua, and J. H. Oh, "Multi-level multi-channel air cache designs for broadcasting in a mobile environment," in *Proc. of IEEE DATA ENGINEERING*, USA, 2000.

[17] T. Imielinski, S. Viswanathan, and B.R. Badrinath, "Energy efficient indexing on air," in *Proc. of ACM SIGMOD*, Minneapolis,MN, May 1994, pp. 25–36.

[18] G. Rawlins, *Compared to What: An introduction to the analysis of algorithms*, Computer Science Press, New York, 1991.

[19] J. C. R. Bennett and H. Zhang, "Hierarchical packet fair queueing algorithm," in *Proc. of ACM SIGCOMM*, 1996.

[20] Stathes Hadjiefthymiades and Lazaros Merakos, "Using proxy cache relocation to accelerate web browsing in wireless/mobile communications," in *Proc. of 10th WWW International Conference*, Hong Kong, May 2001.

[21] G. K. Zipf, *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*, Addison-Wesley, Reading, Mass., 1949.