A Generalized Design for Broadcasting on Multiple Physical-Channel Air-Cache

Duc A. Tran
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2362, U.S.A
dtran@cs.ucf.edu

Kien A. Hua
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2362, U.S.A
kienhua@cs.ucf.edu

Ning Jiang
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816-2362, U.S.A
njiang@cs.ucf.edu

Keywords
Data broadcast, air cache, mobile computing, wireless network

ABSTRACT
In limited and asymmetric bandwidth environments such as wireless networks, push technology can be used to allow a large number of mobile users to access the shared data. Most of the today's designs 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. Specifically, we propose a novel scheduling scheme called *Generalized Broadcast Technique* (GBT) which can adapt the content of the broadcasts according to the client mobility and demand patterns. Theoretical foundations and performance evaluation are also provided to substantiate the near-optimality and to assess the advantages of GBT.

1. INTRODUCTION
In recent years, mobile computing has attracted much attention due to its appealing computing environment [9]. However the narrow bandwidth of wireless networks, and the relatively short active life of power supplies (or battery) 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 technology* [10] 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 to them. They treat air 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 which data and when to send to users in the absence of specific requests. Researchers have proposed algorithms for designing broadcast schedules. These solutions use pure push as in *Broadcast Disks* [1, 2, 3, 5, 6] and *hybrid* techniques (where most frequently accessed items are broadcasts and the others are provided on demand) as in [4, 14, 16]. Other algorithms for scheduling broadcasts are dealt in [7, 8, 11, 12, 15].

Those methods are based on the assumption that there exists a single physical channel for data broadcast. However, there are many scenarios [15] where a server has access to multiple low-bandwidth physical channels, which cannot be combined to form a single high-bandwidth one. The first solution devoting to such problems has been reported in [15] which provided a wide range of design considerations for the server which broadcasts periodically over multiple physical channel air-cache, and for the mobile users to influence the broadcast strategy for their benefit. However, [15] assumed the channels to be homogeneous and data items delivered to be equally sized. In this paper, we study the general case where the air-cache consists of channels with different bandwidths and data items of different sizes. To the best of our knowledge, no previous research on this practical extension has been done. The issues we address include investigating what strategy should be used to partition data over multiple heterogeneous physical channels and what ordering should be used to periodically disseminate data on each channel. We also discuss how the broadcast structures should be changed according to the user retrieval behavior. Especially, we propose a near-optimal scheduling scheme called *Generalized Broadcast Technique* (GBT) to answer the above particular questions. Our mathematical proofs and experimental results justify the correctness and potential of the proposed technique.

The remainder of this paper is organized as follows. The details of GBT are introduced in section 2. Our performance

study is reported in section 3. In Finally in Section 4, we give concluding remarks and provide pointers to future work.

2. MULTI-CHANNEL BROADCASTING

2.1 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. Those channels are physical and may have different bandwidths. We denote by m the number of channels which are indexed as channel 1, channel 2, .., channel m. Assume that channel i has bandwidth B_i and $B_1 \leq B_2 \leq .. \leq B_m$. The objects broadcast by the server are organized as self-identifying "data items" which may be of various sizes. However, techniques to be presented can also be work for other data structure organization. Let n be the number of data items. "Hot" items are defined as those accessed frequently while "cold" ones are accessed less often. The access frequency of an item is called the "temperature". In our adaptive model, temperatures are updated as time goes by and the broadcast content is determined accordingly to the changes. The length and temperature of an item i are expressed by $Length(i)$ and $Temp(i)$, respectively. Furthermore, we have $\sum_{i=1}^{n} Temp(i) = 1$.

We use push-periodic as the delivery mechanism. In other words, each channel is subdivided into logical units called "broadcast units" or "cycles" in short. Data items in those cycles are sent repeatedly over a channel but none is broadcast on more than one channel. In this work, we are interested in optimizing the usage of non-combinable channels; hence for simplicity, we assume that there are enough on-demand channels for the mobile users to send requests to the server. In addition, the users are then informed by the server of which channel to tune in for the data item they request.

We are interested in the following metrics: (1) Mean Waiting Time (MWT): The average delay from when the user requests data to when the user starts receiving it. (2) Mean Active Time (MAT): The average delay from when the user requests data to when the user finishes receiving it. (3) Mean Download Time (MDT): The average time the user needs to download an item. In this section, we will discuss broadcast techniques that are aimed at minimizing MWT and MAT. Since MAT = MWT + MDT, in order to minimize MAT, we build mathematical foundations for the MDT optimization and design algorithms based on that. Essentially, each technique consists of three components: (1) A procedure to allocate data items over multiple channels on which the items will be sent; (2) A procedure to determine the order of disseminating items on each channel; (3) A procedure to update the temperature values of data items based on the user retrieval behaviors.

2.2 Basic technique

Presented in [15] is a scheduling strategy which works very well in the case of homogeneous channels and equally-sized data items. The basic idea is to allocate data items over the channels in such a way that hot items are broadcast more frequently than the cold data items. First, all items are sorted in the non-increasing order of temperature values and stored in a queue *List*. Then, we assign them over the channels by basing on the concept of *Bin-Packing* [13]. It regards each broadcast unit as a *bin* and tries to fill in data items based on their current temperature values. Let $Temp_{avg} = (\sum_{i=1}^{n} Temp(item_i))/m$ where $\{item_1, item_2, .., item_n\}$ is the set of all items. We start from the fastest channel and allocate items in the list *List* until the total temperature of those exceeds $Temp_{avg}$. We then examine the next fastest channel and repeat the strategy. This process stops once all items have been assigned (or until queue L is empty). The details are listed below:

ALGORITHM 2.1. *[Basic Technique]*

1. *Let List be the ordered list of items* $\{item_1, item_2, .., item_n\}$ *such that* $Temp(item_1) \geq Temp(item_2) \geq .. \geq Temp(item_n)$.
2. *Let* $Channel = m$, $i = 1$, $Variable = 0$
3. *Allocate the* i^{th} *element of List to channel Channel. If*

   $$Variable + Temp(item_i) < (\sum_{i=1}^{n} Temp(item_i))/m$$

   then increment i, increase Variable by $Temp(item_i)$, *and repeat this step. Otherwise, go to the next step.*
4. *Decrement Channel, increment i, reset* $Variable = 0$. *If* $Channel > 0$ *and* $i \leq n$, *go back to the previous step.*
5. *Exit.*

In any cycle on each channel, items are delivered in the non-increasing order of $Temp(.)$ *values.*

2.3 Generalized Single Item-Instance Broadcast Technique

The basic technique (BT) was designed for the case with data items of the same size and channels having the same bandwidth. It may not be suitable when that premise does not hold. Taking into account the differences between item sizes and between bandwidths likely improves the effectiveness of the broadcast. In this subsection, we introduce a new technique which is extended from BT to accommodate the above requirements. We call this new approach the *Generalized Broadcast Technique* (GBT).

Let us denote a schedule by $S = \{S_1, S_2, .., S_m\}$ where each $S_i = (item_{i1}, item_{i2}, .., item_{ik_i})$ represents the items broadcast on channel i in that order ($\sum_{i=1}^{m} k_i = n$). We have the average download time resulted by schedule S is $\text{MDT}(S) = \sum_{i=1}^{m} (\sum_{j=1}^{k_i} Temp(item_{ij}) \times Length(item_{ij})/B_i)$. The theorems below give a theoretical basis for the proposed scheme:

THEOREM 2.1. *[GBT-MDT-Minimization] Given S the optimal schedule in terms of MDT, for every pair of items* $(item_{ij}, item_{kl})$ *where* $i < k$, *one of the following must be true:*

1. $B_i = B_k$, *or*

2. $Temp(item_{ij}) \times Length(item_{ij}) \leq Temp(item_{kl}) \times Length(item_{kl})$.

PROOF. By way of contradiction, suppose that $Temp(item_{ij}) \times Length(item_{ij}) > Temp(item_{kl}) \times Length(item_{kl})$ and $B_i < B_k$. We consider a new schedule S' that is similar to S except that $item_{ij}$ and $item_{kl}$ are broadcast on channel k and i, respectively. We have: $\Delta(MDT) = MDT(S') - MDT(S) = Temp(item_{kl}) \times Length(item_{kl})/B_I - Temp(item_{ij}) \times Length(item_{ij})/B_I + Temp(item_{ij}) \times Length(item_{ij}) / B_k - Temp(item_{kl}) \times Length(item_{kl})/B_k \Rightarrow \Delta(MDT) = (Temp(item_{kl}) \times Length(item_{kl}) - Temp(item_{ij}) \times Length(item_{ij})) \times (1/B_i - 1/B_k)$. $B_i < B_k$ implies $1/B_i - 1/B_k > 0$. In association with $Temp(item_{ij}) \times Length(item_{ij}) > Temp(item_{kl}) \times Length\ (item_{kl})$, we have Δ(MDT) < 0. Therefore schedule S' is strictly "better" than S conflicting with the fact that S is the MDT-optimal solution. □

THEOREM 2.2. *[GBT-MWT-Minimization] Given S the optimal schedule in terms of MWT, for every pair of items $(item_{ij}, item_{il})$ where $j < l$, then the following must be true: $Temp(item_{ij})/Length(item_{ij}) \geq Temp(item_{il})$ / Length $(item_{il})$*

PROOF. By way of contradiction, suppose that there exist a pair $(item_{ij}, item_{il})$ so that $j < l$ and $Temp(item_{ij})$ / $Length(item_{ij}) < Temp(item_{il})/Length(item_{il})$. It is always possible to choose j and l such that $l = j+1$. We consider a new schedule S' that is obtained from S by switching the order of $item_{ij}$ and $item_{il}$ in the cycle. Let a user U request an item at some time t. Let $t' \geq t$ be the earliest point of time right before $item_{ij}$ is delivered. If we follow schedule S, the average time U has to wait until starting to receive data is $W(U) = W + (t'-t) \times Temp(item_{ij}) + (t'-t+Length(item_{ij})/B_i) \times Temp(item_{il})$ where W is the average waiting time for items rather than $item_{ij}$ and $item_{il}$. If we follow S', the corresponding time is $W'(U) = W + (t'-t) \times Temp(item_{il}) + (t'-t+Length(item_{il})/B_i) \times Temp(item_{ij})$. Therefore, $W(U) - W'(U) = (Length(item_{ij}) \times Temp(item_{il}) - Length(item_{il}) \times Temp(item_{ij}))/B_i = (Length(item_{ij})$ / $Temp(item_{ij}) - (Length(item_{il})/Temp(item_{il}))/(B_i \times Temp(item_{ij}) \times Temp(item_{il})) > 0$. This inequality holds for any instant of time t, therefore we can say that S' is better than S, which is contradictory since S is the MWT-optimal schedule. □

The GBT-MWT-Minimization theorem suggests a way to order items in each broadcast cycle of a channel while the GBT-MDT-Minimization suggests a way to distribute items over all channels. The strategy is as follows:

ALGORITHM 2.2. *[GBT-Scheduler]*

- *Step 1: Sort all data items in the non-increasing order of the multiplication of temperature and length (MTL).*
- *Step 2: Assign those items with higher MTL values to channels having larger bandwidths.*
- *Step 3: For each channel, sort the items assigned to it in the non-increasing order of the ratio between temperature to length. The ordered list is the final schedule for that channel.*
- *Step 4: Exit.*

There are many ways of designing Step 2. We note that, in the worst case, until receiving a request from a channel C, the user has to wait as long as the time to transmit the entire broadcast cycle of C. It is therefore preferred to minimize the time to deliver the longest cycle. That is, we need to find a schedule S that minimizes $LongestCycle(S) = max_{1 \leq i \leq m} (\sum_{j=1}^{k_i} Length(item_{ij})/B_i)$. Let $l_i = \sum_{j=1}^{k_i} Length(item_{ij})$ and $L = \sum_{i=1}^{m} l_i$. Then $LongestCycle(S) = max_{1 \leq i \leq m} l_i/B_i$. Its minimization should imply that $l_1/B_1 = l_2/B_2 = ... = l_m/B_m = (l_1 + l_2 + .. l_m)/(B_1 + B_2 + .. B_m) = L/\sum_{i=1}^{m} B_i$. In other words, the broadcast length of a cycle on each channel is proportional to the channel bandwidth. We present the algorithm of Step 2, which approximates the above computations, below:

ALGORITHM 2.3. *[GBT-Dispatcher]*

1. *Let List be the ordered list of items as the result of Step 1 of GBT-Scheduler and L the total length of all items in the database.*
2. *Let Channel = m, i = 1, Variable = 0*
3. *Allocate the i^{th} element of List to channel* Channel. *If $Variable + Length(item_i) < B_{Channel} \times L / \sum_{i=1}^{m} B_i$, then increment i, increase Variable by $Length(item_i)$, and repeat this step. Otherwise, go to the next step.*
4. *Decrement Channel, increment i, reset Variable = 0. If Channel > 0 and $i \leq n$, go back to the previous step.*
5. *Exit.*

EXAMPLE 2.1. Let us see an example of a schedule determined by GBT. We have three channels {1, 2, 3} with B_1 = 2 (data units/second), B_2 = 5 (data units/second) and B_3 = 7 (data units/second). There are 6 data items in the database {"A", "B", "C", "D", "E", "F} with $Length$("A") = 3, $Length$("B") = 2, $Length$("C") = 1, $Length$("D") = 5, $Length$("E") = 4, and $Length$("F") = 6. Therefore, L = 21. Suppose that before the server runs its scheduler and dispatcher, we have the temperature values $Temp$("A") = 100, $Temp$("B") = 50, $Temp$("C") = 40, $Temp$("D") = 35, $Temp$("E") = 30, and $Temp$("F") = 20. By GBT-Dispatcher algorithm, items {"A", "D", "E"} will be broadcast on channel 3, {"B", "F} on channel 2, and "C" on channel 1. The order of delivering them within a cycle, which is the result of GBT-Scheduler, is as follows: "A" before "E", "E" before "D", and "B" before "F". The final schedule for this particular adaptive period is illustrated in Figure 1.

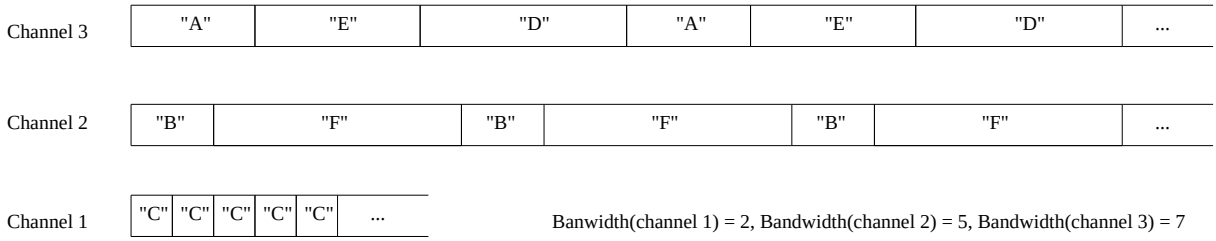

Figure 1: Example of schedules determined by GBT

2.4 Adaptation Issues

We have presented the first two components (dispatcher and scheduler) of each technique. Now we need to work out how the techniques are implemented to adapt changes in user retrieval behavior. The content of each channel's broadcast unit is determined by knowing a priori the data items' length and temperature. Temperature values change over time as the result of user dynamic behavior. In our system, the server runs the dispatcher and scheduler procedure periodically; specifically in every $PERIOD$ units of time.

Let $Temp(1)$, $Temp(2)$, .., $Temp(n)$ be the current temperatures of n data items in the database. Temperatures are reset to 0 at the start of the system. Whenever a request for a data item i arrives at the server, $Temp(i)$ is increased by a constant $HEATING\_FACTOR$. Then the temperatures are normalized so that they add up to 1. That is, $\forall\ j = 1$ to n, $Temp(j)$ is scaled to $Temp(j)/\sum_{k=1}^{n} Temp(k)$. It is also reasonable to reduce the temperature of those items which are not accessed frequently. This task is called the "cooler" procedure which is done periodically. For each item, if its temperature is less than a constant $COOLING\_FACTOR$, the temperature will be initialized to 0; otherwise, the temperature will be decreased by $COOLING\_FACTOR$.

3. PERFORMANCE STUDY

In this section, we present simulation results for aforementioned techniques. We used MWT (Mean Waiting Time), MAT (Mean Active Time) and MDT (Mean Download Time), which are defined in Section 2, as the performance measures. Since the behavior of MAT in our experiments is quite similar to that of MWT, we only report results for MWT and MDT.

3.1 Experimental environment

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 5 Mbytes. To model the information retrieval pattern, data items are requested according to a Zipf-like distribution with skew factor z. Our workload generator creates the service requests according to a Poisson process where the arrival request rate was fixed at 300 requests/s, which is enough to show the reasonable comparisons among the data delivery schemes. We did not investigate the effect of this rate because of the broadcast feature of our delivery system where the data access frequency is already determined by the skew factor above. We assume that the server has multiple broadcast channels having bandwidths which vary from 32Kbytes/s to 100Kbytes/s. 32Kbytes/s models the capacity of a typical in-house wireless network and 100Kbytes/s is in the same order as what is available over a cable modem. Each simulation run lasted one hour. The simulation time unit is a second. In order for the server to accommodate the changes of data item temperatures, the broadcast content was re-calculated every 20 seconds. Furthermore, we set PERIOD = 20 seconds, HEATING_FACTOR = 1, and COOLING_FACTOR = 1. We assume that the time it takes to process this task is ignored. Table 1 summarizes the workload parameters. The detailed numerical results are reported in the following subsections.

Table 1: Parameters

Parameter	default	variation
Database size (items)	1,000	500-1500
Number of channels	30	5-50
Simulation run time (seconds)	3,600	N/A
Skew factor	0.7	0.1-1.0
Request rate (requests/second)	300	N/A

3.2 Effect of data access pattern

To investigate the impact of the skew factor, the arrival rate is set at 300 requests/second. The skew factor varies from 0.1 (i.e., 54% of total requests will go for a particular 56% subset of the data items) to 1.0 (i.e., 90% of total requests will go for a particular 10% subset of the data items). The corresponding MWT and MDT values of each broadcast technique are plotted in Figure 2. The behaviors of GBT and BT are quite similar. However in any simulation run (Figure 2-left), GBT always outperforms the other by a difference of about 100 seconds in terms of waiting time. Figure 2-right illustrates the MDT values. Both techniques provide shorter average download time as the access patter is more skewed. The gap between them gets broader as the skew factor decreases. At any point of scale, GBT is significantly better than BT.

3.3 Effect of number of channels

In this study, each channel has bandwidth between 32Kbytes / s and 100Kbytes / s. Figure 3 shows the performances of BT and GBT under the changes of server capacity. It is understandable that the increase of the number of channels imply the decrease of MWT value. When the server has only 5 channels (limited capacity), each BT-user often spends 6000 seconds until it starts receiving data, but GBT spends 4000 (1.5 times better) seconds. In the case where the server has the most capacity (number of channels equals 50), the MWT values of BT and GBT are about 270 and 450 seconds,

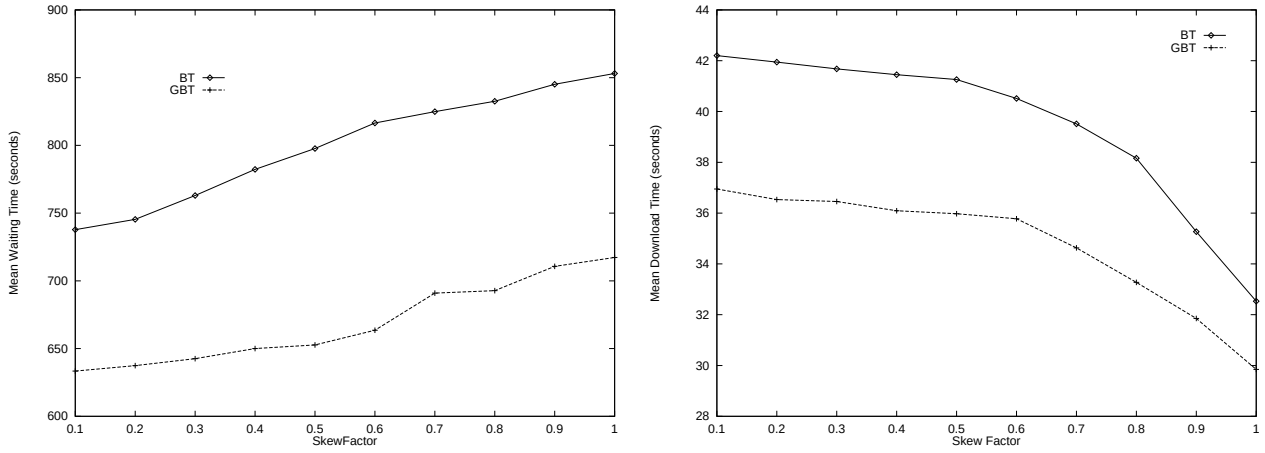

Figure 2: Effect of access pattern

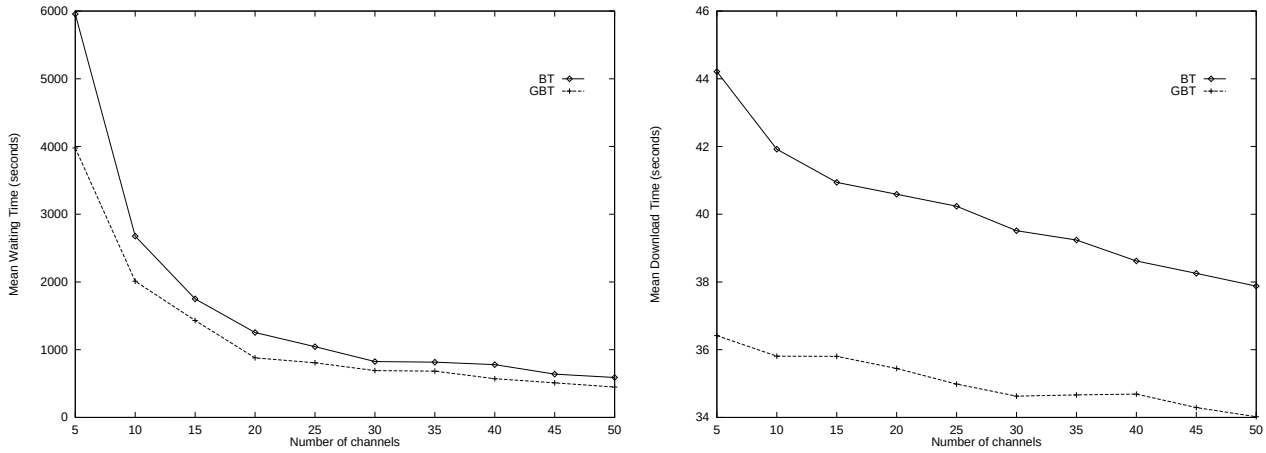

Figure 3: Effect of number of channels

respectively. This exhibits significant improvements of GBT over BT. In terms of MDT values (Figure 3-right), GBT are still superior to BT. The performance gaps among them become clearer as the server has fewer channels. Though, when the server has as large as 50 channels, GBT's MDT value is still small compared to that of BT (i.e., 85% less).

3.4 Effect of database size

We studied the impact of database size on the three techniques by changing the number of items to be broadcast from 500 to 1500 items. The size of any item is generated randomly and can be up to 5Mbytes. In any run, BT always requires the longest average waiting and download time. We can see in Figure 4-left that the two curves are formed in a linear manner with positive slopes. As the number of items increases, the MWT value of GBT gets larger slowly. On the contrary, the MWT grows very quickly in the case of BT. The behaviors of BT, and GBT in terms of MDT (Figure 4-right) are quite similar. The change of database size does not affect much on the download time, but a substantial enhancement is obtained by GBT in comparison with BT. In the default experimental set up (i.e., the number of items is 1000), it takes BT-users about 40 seconds to download an item while GBT-users spend only 34 seconds (85% less).

4. CONCLUSIONS

Previous data broadcast methods assume the model in which the server delivers information on only one physical channel. However, there are many practical situations where the server has access to multiple physical channels but they cannot be combined to form a single high-bandwidth one. In this paper, we have presented a novel technique called Generalized Broadcast Technique (GBT). GBT was designed for the model where no item is broadcast more than once in each broadcast cycle. It is aimed at optimizing the mobile users' waiting time and active time. The near-optimality of this technique is supported by mathematical proofs given in the paper. Its performance advantages are assessed by a detailed simulation model which compared GBT with another technique called the Basic Technique (BT). The experimental results have shown that GBT is superior to BT by a significant margin. As an example, under most scenarios GBT-users need to wait 1.5 times shorter than BT-users. To download an item, on the average, the former spends 85% shorter than the latter.

A major contribution of this work is that the proposed technique is the first to address the *general* problem of utilizing air caches over multiple physical channels. We are looking at

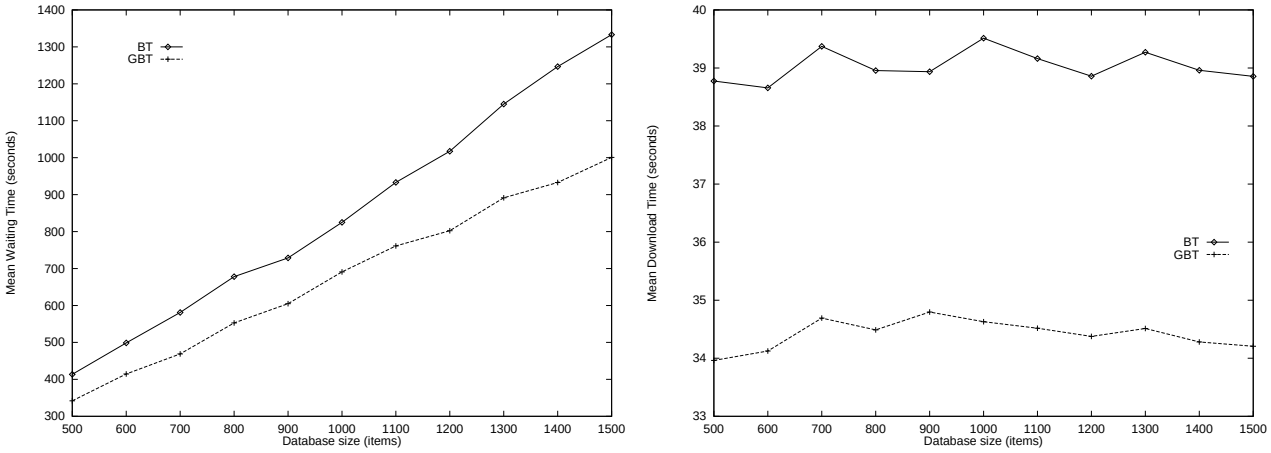

Figure 4: Effect of database size

ways in which we can use GBT to broadcast real-time data efficiently. Additionally, it would be useful to evaluate techniques for broadcasting index information more efficiently. We are also interested in extending GBT so that it works efficiently in the case where items can appear multiple times in a broadcast cycle.

5. REFERENCES

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

[2] S. Acharya, M. Franklin, and S. Zdonik. Disseminating updates on broadcast disks. In *Proc. of VLDB*, pages 354–365, September 1996.

[3] S. Acharya, M. Franklin, and S. Zdonik. Prefetching from a broadcast disk. In *Proc. of IEEE DATA ENGINEERING*, pages 276–285, New Orleans, USA, September 1996.

[4] S. Acharya, M. Franklin, and S. Zdonik. Balancing push and pull for data broadcast. In *Proc. of 1997 ACM SIGMOD*, pages 183–194, May 1997.

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

[6] 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*, pages 543–551, Birmingham, U.K., April 1997.

[7] 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*, pages 124–133, Birmingham, U.K., April 1997.

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

[9] M. H. Dunham and A. Helal. Mobile computing and databases: Anything new? *SIGMOD RECORD*, 24(4):5–9, December 1995.

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

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

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

[13] D. Johnson, A. Demers, J. Ullman, M. Garey, and M. Graham. Performance bounds for simple one-dimensional bin packing algorithms. *SIAM Journal on Computing*, 3(4):299–325, 1974.

[14] 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.

[15] 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.

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