Scheduling for Efficient Data Broadcast over Two Channels

Kevin Foltz
California Institute of Technology
Pasadena, CA 91125
Email: kfoltz@paradise.caltech.edu

Lihao Xu
Department of Computer Science
Washington University
St. Louis, MO 63130
Email: lihao@cs.wustl.edu

Jehoshua Bruck
Department of Electrical Engineering
California Institute of Technology
Pasadena, CA 91125
Email: bruck@paradise.caltech.edu

***Abstract*— The broadcast disk provides a way to distribute data to many clients simultaneously. A central server fixes a set of data and a schedule for sending it, and then repeatedly sends the data according to the schedule. Clients listen for data until it is broadcast. We look at the problem of scheduling for two separate channels, where each can have a different broadcast schedule. Our metric for measuring schedule performance is expected delivery time (EDT), the expected value of the total elapsed time between when a client starts listening for data and when the client is completely finished receiving the data. We fix the first channel with a schedule that is optimal for an average case, and look at how to schedule for the second channel. We show two interesting results for sending two items over two channels. The first is that all schedules with equal portions of the two items in the second channel have the same EDT. The second is that for a situation that is symmetric in the two items the optimal schedule is asymmetric with respect to these items.**

I. Introduction

As wireless computer networks grow more popular, we are faced with the problem of providing scalable, high-bandwidth service to a growing number of users. Wired networks typically use "data pull," where users send requests to a server and the server responds with the desired information. In the wireless domain, "data push" promises to provide better performance for many applications [1]. The broadcast domain that is typical of wireless communication is very effective in distributing information to large audiences.

The idea of broadcast disks has been around since the Teletext system [2]. There is now an interest in applying these ideas to wireless computer networks. Computing optimal schedules has been shown to be difficult [6]. The optimal schedules themselves, however, seem to be less complex, and often periodic.

Work has been done to schedule data broadcast from a server to many clients. However, little of it has looked at methods for more than one channel.

Vaidya and Hameed have looked at multi-channel scheduling [8], but they don't consider combining of information from the two channels. The multi-channel situation would arise, for example, when there are different types of receivers, some with better receiving capabilities than others. One broadcast could take place over a reliable channel that all clients can receive, providing a baseline quality of service. A second broadcast could be sent over a channel that is available only to some of the clients, due to geography, power, or financial contraints. We will examine this problem and give some results concerning scheduling on two different channels.

In Section II, we describe our two-channel broadcast model and the corresponding scheduling problem. In Section III, we look at some specific two-channel schedules, where the schedule for channel 1 is fixed and the schedule for channel 2 is constrained to have equal numbers of packets of each item. We show that all such schedules give the same performance. In Section IV we show that we can find asymmetric schedules that are better than any symmetric schedule even when the scheduling problem is symmetric with respect to the data items. Section V presents conclusions and identifies areas for future research.

II. Broadcast Model and Problem

Our model consists of two servers broadcasting two data items to many clients. We assume that each item is broken into a large number of packets, which can be received independently, so that a client can start receiving data in the middle of a transmission of a data item. Each server has a broadcast channel of fixed constant bandwidth B, and we assume $B = 1$ for each channel. We assume each data item has the same length $l = 1$. Server 1 broadcasts on channel 1, sending item 1 from time 0 to 1, and item 2 from time 1 to 2. Server 1 then repeats this, alternately sending items 1 and 2.

Delivery time (DT) is the length of time between when a client first starts listening for a data item and when it completes reception of the entire data item. Given a fixed broadcast schedule, the DT for an item depends on the instant in time a client starts waiting. For example, at time 0, the DT for item 1 (DT_1) will be 1, as shown in Figure 1a, since the client simply listens while server 1 sends the item. However, at time $\frac{3}{2}$, DT_1 will be $\frac{3}{2}$, as in Figure 1b, since the client must wait through half of item 2 and then item 1, for a total wait of $\frac{1}{2}$ + 1 = $\frac{3}{2}$. So, at time $\frac{1}{2}$, DT_1 is 2, as in Figure 1c, since the client receives half of item 1, then waits while item 2 is sent, and then receives the other half of item 1, for a total delivery time of $\frac{1}{2} + 1 + \frac{1}{2} = 2$.

The expected delivery time for item 1 (EDT_1) is simply the expected value of DT_1 with respect to all possible initial listening times of a client. We assume broadcasts are periodic,

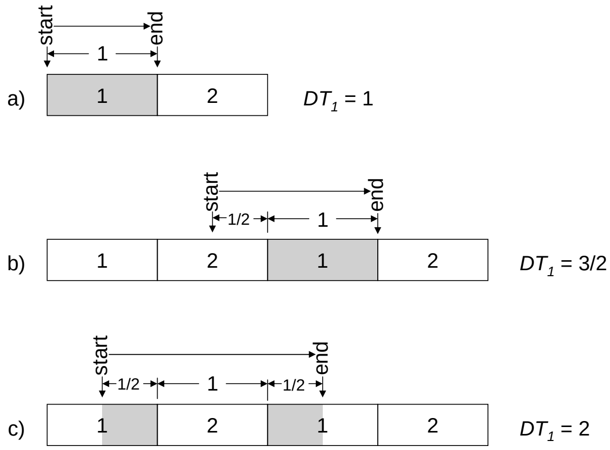

Fig. 1. Computing DT_1 for one channel at different points in the schedule 12. The gray areas indicate where useful data is being received by the client. a) starting at time 0, $DT_1 = 1$, b) starting at time $\frac{3}{2}$, $DT_1 = \frac{3}{2}$, c) starting at time $\frac{1}{2}$, $DT_1 = 2$

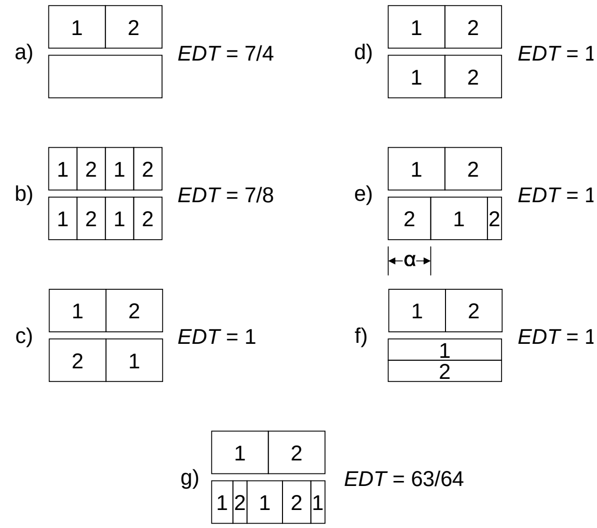

Fig. 2. Scheduling for two channels with $p_1 = p_2 = \frac{1}{2}$. a) Optimal schedule for one channel, b) Optimal schedule for two channels, without channel 1 being fixed, c) Constant-DT schedule, d) Repetition schedule, e) Repetition schedule with a shift by α, f) Splitting the second channel into sub-channels, g) Asymmetric schedule that is better than all symmetric schedules

so in computing EDT_i it is sufficient to compute EDT_i over one period. We assume each data item has an aggregate demand by the clients. We represent these demands by demand probabilities p_1 and p_2 for item 1 and item 2, respectively. We assume client requests are Poisson with rates proportional to p_1 and p_2, where p_1 and p_2 are normalized to sum to 1. For example, if item 1 is requested at twice the rate of item 2, then $p_1 = \frac{2}{3}$ and $p_2 = \frac{1}{3}$. To compute expected delivery time (EDT), we weight EDT_1 and EDT_2 by their demand probabilites, so $EDT = p_1 \cdot EDT_1 + p_2 \cdot EDT_2$. We will use EDT to evaluate broadcast schedule performance. The lower the value of EDT, the better a schedule is. Intuitively, EDT represents the average length of time clients wait for items, and a schedule that gives the shortest average wait is an optimal schedule.

We write schedules by writing the numbers of the items broadcast, with exponents to denote the length of time each item is sent. For example, alternating items 1 and 2 gives the schedule $1^1 2^1$, or 12 (exponents are assumed to be 1 when omitted). Sending half of item 1 and half of item 2 alternately would be written as $1^{\frac{1}{2}} 2^{\frac{1}{2}}$. Since broacasts are cyclic, the schedules 1212 and 12 represent the same broadcast schedule. Also $1^{\frac{1}{2}} 2^{\frac{1}{2}} 1^{\frac{1}{2}} 2^{\frac{1}{2}}$ and $1^{\frac{1}{2}} 2^{\frac{1}{2}}$ represent the same broadcast schedule. In general, $1^{\frac{1}{k}} 2^{\frac{1}{k}}$ means to partition 1 and 2 into k equal-size subitems, and then send those subitems alternately.

Error correcting codes are very effective in maintaining performance in the presence of packet losses. [5]. Error correcting codes are also very useful in scheduling over multiple channels. An (n, k) code encodes a k-symbol message to an n-symbol codeword such that the original k message symbols can be recovered from *any* m symbols of the codeword, where $k \leq m \leq n$. A symbol is a flexible data unit, such as a byte, a frame, or a packet. When $m = k$, the code is called an MDS (Maximum Distance Separable) code [7].

We apply an (n, k) MDS code on a data item, where n is a multiple of k, e.g., $n = 2k$. The MDS property ensures that any k symbols of the encoded data item can be used to recover the original data item. In particular, if we partition the encoded data item into two equal halves, each with k symbols, then either half can be used to recover the original data item. Thus the two halves can be used alternately in schedules to replace the original data item. This makes scheduling over multiple channels much more flexible, as complex synchronizations are not necessary to fully utilize the aggregated bandwidth of multiple channels. A nice feature of such coding is that there is no price to pay in performance.

From previous work [3], we find that the optimal schedule for $p_1 = p_2 = \frac{1}{2}$ is simply the schedule 12. This schedule also has the nice property of being optimal, in a restricted sense, for values of p_1 between $\frac{3}{8}$ and $\frac{5}{8}$, and has constant EDT for all values of p_1. For these reasons, we fix the schedule for channel 1 as 12 and attempt to find optimal schedules for channel 2, given this schedule for channel 1.

For two channels, the EDT is computed the same way as for a single channel, except now it is possible to get more than one packet of each item concurrently. As a result, the EDT values for two channels are lower than those for either of the individual channels alone. We can think of two channels as simply being one channel with twice the bandwidth. However, in this work we think of the two channels as being distinct, since some clients may only be able to access the first channel.

III. Schedules with Equal Amounts of Each Item

We consider $p_1 = p_2 = \frac{1}{2}$ with schedule 12 on the first channel, giving $EDT = \frac{7}{4}$. This is shown in Figure 2a. In the figure, we represent schedules in time and bandwidth, with time going horizontally (left to right is forward in time) and bandwidth expressed vertically (the height of a block indicates the bandwidth over which it is broadcast). In each schedule, we show time from 0 to 2 and bandwidth split into channel 1 (top part) and channel 2 (bottom part). For the first schedule, as in Figure 2 a, channel 2 is unused, and $EDT = \frac{7}{4}$. The best we can do for two channels would be half the optimal time for one channel, or $EDT = \frac{7}{8}$. This is achieved using schedule $1^{\frac{1}{2}}2^{\frac{1}{2}}$ on each channel, as shown in Figure 2b. However, we cannot achieve this lower bound when restricted to keep the schedule 12 on channel 1. We are interested in how close we can get to this bound, with channel 1 restricted.

One simple approach is to broadcast data using schedule 21 on channel 2, as in Figure 2c, guaranteeing that at any time each item is being sent, for an easily-computed EDT of 1. Another approach is to use the same schedule, 12, as in Figure 2d. The EDT for this case as 1, also. In general, we can consider all schedules that are simple offsets of 12, as in Figure 2e. We consider offsets α from 0 to 2 and find that for any offset α the corresponding EDT is 1. A different approach would be to split channel 2 into two half-bandwidth channels, sending item 1 on the first and item 2 on the second, as in Figure 2f. This also has $EDT = 1$.

We now present a more general theorem that includes these cases.

Theorem 1: Given two channels, each of bandwidth $B = 1$, two items of length $l = 1$, and broadcast schedule for channel 1 fixed at 12. Any periodic schedule for channel 2 with period 2 and equal amounts of items 1 and 2 gives a two-channel EDT of 1.

Proof of Theorem 1:
Since everything is symmetric with respect to the items, we consider item 1 WLOG. We know that the expected waiting time, and hence EDT, is the same if we reverse the schedules with respect to time [4]. At any point in the schedule we can compute both a forward and backward delivery time for item 1, $FDT_1 (= DT_1)$ and BDT_1, respectively. The averages of FDT_1 and BDT_1 over the entire schedule are the same, since $EFDT_1 = EDT_1 = EBDT_1$, where $EFDT_1$ and $EBDT_1$ are computed from FDT_1 and BDT_1 the same way that EDT_1 is computed from DT_1.

We compute $FDT_1 + BDT_1$ for all possible starting times. We consider first all starting times such that at the forward ending time (the time at which we finish receiving data item 1 in the forward direction) item 1 is in mid-transmission (i.e. item 1 continues to be broadcast immediately after the ending time).

Any starting time with forward ending time in mid-transmission will have the following condition met:

$$FWT_1 + BWT_1 = 2$$

We can see this by looking in the reverse-time direction. If our forward ending time is in mid-transmission, our reverse ending time will be, too, since the total amount of item 1 in the two channels during one period is 2. This is illustrated in Figure 3 a. Note that this holds because we are in mid-transmission, guaranteeing that when we count backwards, we must end at the same place. So, we have $EFDT_1 = EBDT_1$ and $EFDT_1 + EBDT_1$ = 2. Together these give us $EFDT_1 =$

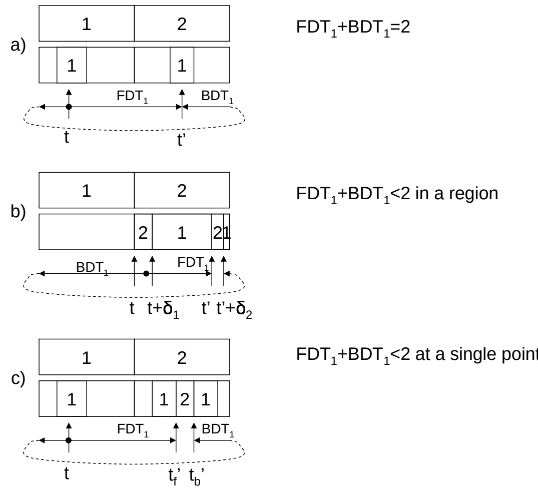

Fig. 3. Forward and Backward Delivery Times for item 1 (FDT_1 and BDT_1). a) If the ending time is in mid-transmission (of item 1), then $FDT_1 + BDT_1 = 2$, b) If neither the starting nor the ending time is in mid-transmission, then $FDT_1 + BDT_1 < 2$ for an interval, c) If the ending time is not in mid-transmission but the starting time is, $FDT_1 + BDT_1 < 2$ at that starting point, but not in an interval around that point.

1, or equivalently $EDT_1 = 1$. Similarly, $EDT_2 = 1$. So, $EDT = p_1 \cdot EDT_1 + p_2 \cdot EDT_2 = 1 \; \forall p_1, p_2$.

We complete the proof by noting that, except for a small number of points, any starting time will give an ending time in mid-transmission. Suppose not. Then we have a range from t to $t + \delta_1$ such that the forward and backward ending times are different. This is shown in Figure 3b, where the forward ending time is t' and the backward ending time is $t' + \delta_2$ for all starting times between t and $t + \delta_1$. This implies we have two blocks of time, t to $t + \delta_1$ and t' to $t' + \delta_2$ (where t' is the ending time for t to $t + \delta_1$) such that only item 2 is sent in each block, and between each block there is exactly one unit of item 1 sent. However, channel 1 is fixed at 12, so one of these blocks must start at $t = 1$ and the other must end at $t = 2$. But then there is no way to send one unit of item 1 between these two times, since there is only $1 - \delta_1 - \delta_2 < 1$ unit of time to send 1 unit of item 1. If follows that there are no ranges of times with non-midtransmission ending times. It follows that EDT is almost always 1. The small number of starting points where we do not end in mid-transmission contribute negligibly to EDT, since the probability of arriving at these points in time is correspondingly small. Figure 3c illustrates a single point, t, where $FDT_1 + BDT_1 < 2$.

It is interesting to note that this result holds even if we split channel 2 into sub-channels. For example, if we divide channel 2 into two channels and broadcast one item on each channel, the result still holds. This result applies to any channel 2 schedule as long as items 1 and 2 are sent in equal amounts (measured as bandwidth integrated over time) per period.

IV. The Asymmetry of Minimizing EDT

One would expect that the optimal schedule for two items of equal length and equal demand would be symmetric with respect to the items. All previously known optimal schedules have this property. However, we show that for two-channel scheduling this is not the case. We do not know the optimal schedule, but we know from the previous section that any symmetric schedule will have $EDT = 1$. We will show asymmetric schedules with $EDT < 1$.

We partition each data item into k equal-sized pieces. We look at $k = 1, 2, 3, ..., 8$ and compute the EDT values for every possible scheduling of these packets on the second channel.

We find the following:

k	Optimal Schedule	EDT	decimal	#1's,2's
1	12	1	1.0000	1,1
2	$1^{\frac{1}{2}}2^{\frac{1}{2}}1^{\frac{1}{2}}2^{\frac{1}{2}}$	1	1.0000	2,2
3	$1^{\frac{1}{3}}2^{\frac{1}{3}}1^{\frac{2}{3}}2^{\frac{1}{3}}1^{\frac{1}{3}}$	95/96	0.9895	4,2
4	$1^{\frac{1}{4}}2^{\frac{1}{4}}1^{\frac{1}{2}}2^{\frac{1}{2}}1^{\frac{1}{2}}$	505/512	0.9863	5,3
5	$1^{\frac{2}{5}}2^{\frac{1}{5}}1^{\frac{3}{5}}2^{\frac{3}{5}}1^{\frac{1}{5}}$	197/200	0.9850	6,4
6	$1^{\frac{1}{3}}2^{\frac{1}{6}}1^{\frac{1}{2}}2^{\frac{2}{3}}1^{\frac{1}{3}}$	379/384	0.9869	7,5
7	$1^{\frac{3}{7}}2^{\frac{2}{7}}1^{\frac{5}{7}}2^{\frac{3}{7}}1^{\frac{1}{7}}$	773/784	0.9859	9,5
8	$1^{\frac{3}{8}}2^{\frac{1}{4}}1^{\frac{5}{8}}2^{\frac{1}{2}}1^{\frac{1}{4}}$	63/64	0.9843	10,6

We note that for $k > 3$, there is always a schedule with $EDT < 1$ and the optimal such schedule has an *unequal* number of packets of each item. Also, the arrangement of these packets seems to follow a somewhat regular pattern.

Even though these particular schedules may not be optimal, the fact that their EDT is less than any symmetric schedule shows that for a scheduling scenario that is completely symmetric in the items, the optimal schedule is asymmetric with respect to these items.

V. Conclusions and Future Work

We have examined broadcast scheduling for two channels. We consider the case where all receivers can access a first channel, and some of them can also receive from a second channel. We fix the schedule for the first channel to give uniform performance, and look at how to optimize the second channel given the first. We show that any second channel schedule that sends equal amounts of each item will perform equally well. We show that we can achieve better performance by sending unequal amounts of each item on the second channel. This is unlike previous results for a single channel, in which all symmetric scheduling problems yielded symmetric schedules.

There are many areas for future research. It would be good to find what the optimal second-channel schedule is. It would also be good to examine what happens for more channels, more items, and items of different lengths. We would like to investigate how performance degrades in the presence of transmission errors, and explore methods to effectively combat errors. Another issue is channel synchronization. We have assumed our two channels are synchronized. It would be useful to examine which schedules perform well when the channels have anywhere from a small degree of alignment error to random alignment.

References

[1] D. Aksoy, M. Altinel, R. Bose, U. Centemel, M. Franklin, J. Wang, S. Zdonik, "Research in Data Broadcast and Dissemination," *Proceedings of 1st International Conference on Advanced Multimedia Content Processing*, Osaka, Japan, Nov. 1998.

[2] M. H. Ammar, J. W. Wong, "On the Optimality of Cyclic Transmission in Teletext Systems," *IEEE Transactions on Communications*," vol. 35, pp. 68-73, January 1987.

[3] K. Foltz, J. Bruck, "Robustness of Time-Division Schedules for Internet Broadcast," ISIT 2002, Lausanne, Switzerland.

[4] K. Foltz, J. Bruck, "Splitting Schedules for Internet Broadcast Communication," *IEEE Trans. Info. Theory*, vol. 48, pp. 345-358, Feb. 2002.

[5] K. Foltz, L. Xu and J. Bruck, "Coding and Scheduling for Loss Resilient Data Broadcasting," ISIT 2003, Yokohama, Japan.

[6] C. Kenyon, N. Schabanel, "The Data Broadcast Problem with Non-Uniform Transmission Times," Proc. of the 10th ACM-SIAM Symp. on Discrete Algorithms (SODA '99), pp. 547-556, January 1999.

[7] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*, Amsterdam: North-Holland, 1977.

[8] N. H. Vaidya, S. Hameed, "Data Broadcast Scheduling: On-line and Off-line Algorithms," Tech. Report 96-017, Computer Science Dept., Texas A&M University, May 1996.