Splitting Schedules for Internet Broadcast Communication

Kevin Foltz Jehoshua Bruck

California Institute of Technology
Mail Code: 136-93
Pasadena, CA 91125

Email: {kfoltz,bruck}@paradise.caltech.edu

Abstract

The broadcast disk provides an effective way to transmit information from a server to many clients. Information is broadcast cyclically and clients pick the information they need out of the broadcast. An example of such a system is a wireless web service where web servers broadcast to browsing clients. Work has been done to schedule the broadcast of information in a way that minimizes the expected waiting time of the clients. This work has treated the information as indivisible blocks. We propose a new way to schedule the broadcast of information, which involves splitting items into smaller pieces that need not be broadcast consecutively. This relaxes the previous restrictions, and allows us to have better schedules with lower expected waiting times. We look at the case of two items of the same length, each split into two halves, and show that we can achieve optimal performance by choosing the appropriate schedule from a small set of schedules. We derive a set of optimal schedules and show which one to use, as a function of the demand probabilities. In fact we prove the surprising result that there are only two possible types of optimal cyclic schedules for items 1 and 2. These start with 1122 and 122122. For example, with demand probabilities $p_1 = .08$ and $p_2 = .92$, the best order to use in broadcasting the halves of items 1 and 2 is a cyclic schedule with cycle 122122222. We also show that much of the analysis remains the same if we consider items of different lengths. We present numerical data that suggests that the set of optimal schedules for different length items also consists of two types, starting with 1122 and 122122. For example, with demand probabilities $p_1 = .08$ and $p_2 = .92$ as above, but $l_2 = 2l_1$, the best schedule is 11222222.

1 Introduction

As mobile computing gains popularity, it becomes increasingly important to find efficient methods of communicating with mobile clients. In general, the mobile clients have considerably less outgoing bandwidth than incoming bandwidth, making communication highly asymmetric. Web browsing is a good example of this situation. A person browsing the web typically receives a lot of information and sends very little. We will describe an efficient way to send information to portable web browsers.

The broadcast disk is a way to send information to many clients at the same time. Using this scheme, data is broadcast through the air in a cyclic fashion. When a client wants some data, it listens to this broadcast stream until it receives the desired data. If the desired data is not in the broadcast cycle, other means are used to retrieve the data. Essentially, the broadcast disk acts as a common cache for many clients, where data in this cache is made available cyclically, according to the broadcast schedule. Our goal is to schedule the broadcast of information in a way that minimizes the expected waiting time of the clients.

Vaidya and Hameed [5, 6, 9] derived the optimal broadcast frequencies of items within a schedule as a function of their demand probabilities, p_i, and lengths, l_i. They showed that to minimize expected waiting time, the frequencies of broadcast, f_i, should be proportional to $\sqrt{\frac{p_i}{l_i}}$. This led to an algorithm that attempted to achieve these relative frequencies. This algorithm is good because it is computationally fairly simple and works for an arbitrary number of broadcast items with arbitrary lengths and demand probabilities.

Jiang and Vaidya [7] discuss ways to minimize the variance of the response time. They also propose a way to trade-off between minimizing the mean and minimizing the variance of the response time. Aksoy and Franklin [1] discuss scheduling the broadcast of information based on client requests. They consider such metrics as average and worst case performance, scheduling overhead, and robustness in the presence of environmental changes. Bestavros [3] describes a way to add fault tolerance to broadcast disks by sending parity information in addition to data. Bar-Noy, Bhatia, Naor, and Schieber [2] look at scheduling in general, and show that there is an optimal cyclic schedule for a broadcast disk, and finding it is NP-hard. Leong and Si [8] discuss how to choose which items to broadcast, using ideas of cache management. Franklin, Zdonik, Acharya, and Alonso [4, 10] also discuss aspects of broadcast disks.

We examine the scheduling of items for a broadcast disk. We represent a broadcast schedule for two items by a sequence of 1's and 2's, where '1' represents the broadcast of the first item, and '2' represents the broadcast of the second item. In this paper, we think of each item as consisting of two halves, and a 1 or 2 will represent one of these halves, not the entire item. For example, a schedule in which the two items are broadcast in their entirety alternately is 112211221122... and not 121212..., since we need two halves of each item to broadcast the entire item.

Broadcast schedules are cyclic, so we will represent them by one of their cycles. Since each item has two halves, we assume these halves are broadcast alternately. The schedule 122, for example, should really be written as $1_A2_A2_B1_B2_A2_B$, where 1_A and 1_B are the two halves of item 1, and 2_A and 2_B are the two halves of item 2. This would more accurately represent one period, but we shorten the representation to 122 with the understanding that the two halves of each item are broadcast alternately. As a shorthand representation, we will sometimes use exponents to indicate repeated items. For example, 12^2 represents 122 and 1^22^312 represents 1122212.

Another representation of a schedule that we will use is based on the number of 2's between consecutive 1's in the schedule. We use a bracketed sequence of numbers that represent the number of 2's between each consecutive pair of 1's. For example, [0,2] represents the schedule 1122, and [0,0,1,3,2] represents 11121222122. We use the notation S^C to represent the complement of S, the schedule S with 1's and 2's swapped. For example, if $S = 12211112$, then $S^C = 21122221$. We use

S^R to represent the reversal of S. With S as above, $S^R = 21111221$.

Sometimes we want to indicate that a certain instance of an item may or may not be present in a schedule. Parentheses will be used to indicate the possible presence of an instance of an item in a schedule. For example, a schedule in which we know only that item 2 is never broadcast twice consecutively could be represented as $1(2)1(2)\ldots1(2)$.

We define a useful function dealing with expected waiting times.

Definition 1 *$EWT(S,p_1)$ is the expected waiting time using schedule S with demand probabilities p_1 and $p_2 = 1 - p_1$, assuming two items, each split into two halves.*

This is our metric for evaluating schedules. The lower the value of EWT(S,p_1), the better schedule S is for demand probabilities p_1 and $p_2 = 1 - p_1$.

Vaidya and Hameed assume a uniform spacing of items within a schedule (which is the optimal spacing, when achievable) to derive their broadcast frequencies. However, in most cases it is not possible to achieve these frequencies with uniform spacing, since the individual schedules for the different items usually do not mesh together perfectly. This can lead to a large deviation of the expected waiting time from the optimal time.

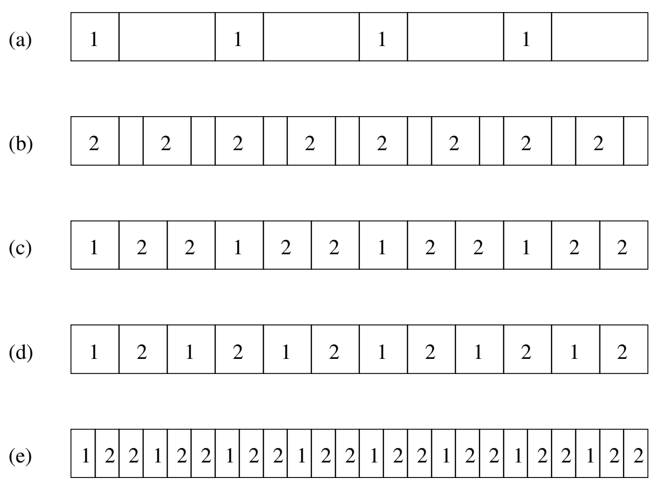

Figure 1: (a) ideal scheduling for item 1, (b) ideal scheduling for item 2, (c) and (d) actual schedules generated using Vaidya and Hameed's algorithm, (e) our schedule using splitting

As an example, consider two items, one that is accessed frequently and another that is accessed less frequently. Suppose $l_1 = 1, p_1 = .2, l_2 = 1, p_2 = .8$. Then, according to Vaidya and Hameed, we want $f_1 = \frac{1}{3}, f_2 = \frac{2}{3}$. We want to broadcast uniformly, so we should schedule items 1 and 2 as in figure 1(a) and (b). If we could somehow merge these two schedules together, we would achieve an expected waiting time of $\frac{9}{10}$. However, it is impossible to merge these schedules together while preserving the property of uniform spacing. The scheduling algorithm described by Vaidya and

Hameed gives us one of the schedules in Figure 1(c) or (d), depending on how we break ties. These have expected waiting times 1 and $\frac{29}{30}$.

To address the problem of merging schedules, we relax the restriction used by Vaidya and Hameed of broadcasting an item in one piece, and look at splitting items into sub-items and scheduling the broadcast of these sub-items. We look at the specific case of two items of the same length, each split into two sub-items. We find that the optimal schedule for the example above is the schedule in Figure 1(e), with expected waiting time $\frac{49}{60}$. Note that this result, when splitting is allowed, is better than even the unachievable theoretical limit of $\frac{9}{10}$ when splitting is not allowed.

In general, we show the following:

Theorem 1 *For two items of the same length, each split into two halves, the broadcast schedule that minimizes expected waiting time is:*

$$1122\overbrace{2\ldots2}^{n}, n = Max\left(0, \left\lceil\frac{-9+\sqrt{-47+\frac{40}{p_1}}}{2}\right\rceil\right),\ \textit{if } p_1 \in \left[\frac{1}{5}, \frac{1}{2}\right]$$

$$122122\overbrace{2\ldots2}^{n}, n = Max\left(0, \left\lceil\frac{-13+\sqrt{-103+\frac{32}{p_1}}}{2}\right\rceil\right),\ \textit{if } p_1 \in \left(0, \frac{1}{5}\right)$$

In Section 2, we present lemmas useful in proving Theorem 1. We discuss the implications of each lemma, leaving their proofs for the appendix. In Section 3, we prove the theorem. We describe how to reduce the search for optimal schedules from all schedules to a smaller set of irreducible schedules. We determine which schedules are in this set. Then, we compare these with each other and see that the schedules listed in Theorem 1 are optimal on their respective intervals.

In Section 4, we consider items of different lengths. We present numerical results that suggest that the optimal schedules for different length items are the same as those for items of the same length. In Section 5, we discuss future directions for research. The appendix contains proofs of the lemmas stated in Section 2.

2 The Lemmas

These Lemmas provide ways of manipulating schedules into similar but better schedules. Their proofs are in the appendix. In each lemma, we assume two items, split into two halves, with $l_1 = 1$.

Lemma 1 *(The Rearrangement Lemma)*

The following hold for all values of p_1:

(a)

$$EWT(S', p_1) \leq EWT(S, p_1),\ \textit{where}$$

$$S = \ldots 1\overbrace{2\ldots2}^{a\geq0}1\overbrace{2\ldots2}^{b\geq3}1\overbrace{2\ldots2}^{c\geq0}1\ldots1\overbrace{2\ldots2}^{d\geq0}1\overbrace{2\ldots2}^{e\geq2}1\overbrace{2\ldots2}^{f\geq0}1\ldots,$$

$$S' = \ldots 1\overbrace{2\ldots2}^{a}1\overbrace{2\ldots2}^{b-1}1\overbrace{2\ldots2}^{c}1\ldots1\overbrace{2\ldots2}^{d}1\overbrace{2\ldots2}^{e+1}1\overbrace{2\ldots2}^{f}1\ldots,\ \textit{and}$$

$$(a+b+c) - (d+e+f) \geq 1$$

(b)

$$EWT(T', p_1) \leq EWT(T, p_1),\ \textit{where}$$

$$T = \ldots 1\overbrace{22\ldots222}^{n\geq2}121\overbrace{2}^{x=0,1}1\overbrace{2}^{y=0,1}1\ldots,$$

$$T' = \ldots 1\overbrace{22\ldots222}^{n}112\overbrace{\;2\;}^{x}1\overbrace{\;2\;}^{y}1\ldots$$

(c)

$$EWT(U', p_1) \leq EWT(U, p_1),\ \textit{where}$$

$$U = \ldots\overbrace{1\ldots1}^{r\geq 1}\overbrace{2\ldots2}^{s\geq 1}\overbrace{11\ldots11}^{t\geq 3}\overbrace{22\ldots22}^{u\geq 3}1\ldots,$$

$$U' = \ldots\overbrace{1\ldots1}^{r}\overbrace{2\ldots2}^{s}\overbrace{11\ldots1}^{t-1}21\overbrace{2\ldots22}^{u-1}1\ldots$$

(d)

$$EWT(V', p_1) = EWT(V, p_1),\ \textit{where}$$

$$V = \overbrace{1\ldots1}^{a_1}\overbrace{2\ldots2}^{b_1}\overbrace{1\ldots1}^{a_2}\overbrace{2\ldots2}^{b_2}\ldots\overbrace{1\ldots1}^{a_{k-1}}\overbrace{2\ldots2}^{b_{k-1}}\overbrace{1\ldots1}^{a_k}\overbrace{2\ldots2}^{b_k},$$

$$V' = \overbrace{2\ldots2}^{b_k}\overbrace{1\ldots1}^{a_k}\overbrace{2\ldots2}^{b_{k-1}}\overbrace{1\ldots1}^{a_{k-1}}\ldots\overbrace{2\ldots2}^{b_2}\overbrace{1\ldots1}^{a_2}\overbrace{2\ldots2}^{b_1}\overbrace{1\ldots1}^{a_1}$$

Lemma 2 *(Corollary to Lemma 1 part (a)) If, instead of $S = [a, b, c, \ldots, d, e, f, \ldots]$, S is one of $[a,\ b,\ c = d,\ e,\ f,\ \ldots]$, $[a,\ b = d,\ c = e,\ f,\ \ldots]$, $[a = f,\ b,\ c = d,\ e]$, or $[a = e,\ b = f,\ c = d]$, then Lemma 1 part (a) still holds, where S' is S with b decremented by 1 and e incremented by 1.*

Part (a) and its corollary tell us that it is generally not good to have strings of 2's of significantly different lengths. More specifically, for each string of two or more 2's, we add its length to the sum of the lengths of the adjacent strings of 2's. These sums should be as close to each other as possible (equal or within 1) for all strings of two or more 2's in the schedule.

For example, for the schedule 12122222112122122222222121, consider the two long strings of 2's of lengths 5 and 7. Their adjacent strings of 2's have lengths 1 & 0 and 2 & 1, respectively, giving sums of 5+1+0=6 and 7+2+1=10. Since $10-6 = 4 \geq 1$, we can move a 2 from the string of length 7 to the string of length 5, giving the new and better schedule 12122222211212212222222121, with sums 6+1+0=7 and 6+2+1=9. Since our new sums differ by 2, we can move another 2 to get an even better schedule 12122222221121221222222121 with sums 8 and 8. However, moving another 2 will not give us a better schedule, since $8-8 = 0 \not\geq 1$.

The corollary allows us to eliminate adjacent long strings of 2's. We do this by performing the operation in the lemma. We call our adjacent strings the b- and e-length strings. After multiple applications of the lemma (move a 2 from one of these strings to the other), we get one of the strings down to length 2.

As an example, consider the schedule 1212222212222112. We have adjacent strings of 2's of lengths 5 and 4. By considering these strings as we did above, we get sums of 5+1+4=10 and 4+5+0=9. So, if we move a 2 from the string of length 5 to the string of length 4, we improve our schedule. However, unlike before, our sums remain the same at 4+1+5=10 and 5+4+0=9, so we can repeat this procedure until we are left with only two 2's in the first string and a new schedule of 1212212222222112.

Part (b) applies when we have blocks of 1's, possibly with some single 2's in them, bounded on both sides by at least two 2's. This lemma tells us that if the beginning or end of the block is 121, it is better to shift the 2 toward the inside of the block to get 112[*rest of block*] or [*rest of block*]211, when there is a total of at least four 1's in the block.

An example of this is the schedule 12212111212112222. This contains the string 1211121211, which is a block of seven 1's with only single 2's, bordered on both sides by 22. Part (b) of the lemma tells us that it is better to have 112 than 121 at the beginning of this string. It is better to rearrange the start of the string to get 1121121211, for a new and better schedule of 12211211212112222.

Part (c) tells us that large strings of 1's and 2's should not border each other. It is better to swap the innermost 1 and 2 if we have at least three of each. A simple example of this is the schedule 121111122212. We would be better off using 121111212212.

Part (d) tells us that reversing a schedule does not affect its expected waiting time. So, the schedules 121112222 and 222211121, for example, have the same expected waiting times.

Lemma 3 *(The Splitting Headache Lemma)*

Suppose schedule S is written as $s_1 s_2 \ldots s_l$, where each s_i is either a 1 or a 2. Suppose also that there is a schedule $C = c_1 c_2 \ldots c_k$, such that C has at least two 1's and at least two 2's, and $c_i = s_{m_1+i \bmod l} = s_{m_2+i \bmod l}$ $\forall i,\ 1 \leq i \leq k$, for some $m_1 \neq m_2 \bmod l$. Then $EWT(S,\ p_1) = \frac{l_1}{l} \cdot EWT(S_1,\ p_1) + \frac{l_2}{l} \cdot EWT(S_2,\ p_1)$, where $S_1 = s_{(m_1+1) \bmod l} s_{(m_1+2) \bmod l} \cdots s_{(m_1+l_1) \bmod l}$, $S_2 = s_{(m_2+1) \bmod l} s_{(m_2+2) \bmod l} \cdots s_{(m_2+l_2) \bmod l}$, $l_1 = (m_2 - m_1) \bmod l$, and $l_2 = (m_1 - m_2) \bmod l$.

This lemma tells us that under certain conditions we can split a schedule S into two schedules, S_1 and S_2. Schedule S will have an expected waiting time that is the weighted mean of the expected waiting times of S_1 and S_2. At any value of p_1, one of S_1 and S_2 will have a shorter expected waiting time and the other will have a longer expected waiting time. As a result, we should never use S, but instead choose the better of S_1 and S_2.

As an example, consider the schedule 11212122221122. We rewrite it (by shifting, since we send data cyclically) as 12121222211221. This is just 12 concatenated with 121222211221. Each of these starts with 1212, which is a string with two 1's and two 2's. It's easy to see this for the second subschedule. For the first one, think of it not as 12, but as 121212..., which is what we broadcast when we use this schedule. We can split our schedule into these two schedules. We can find the point p where the two schedules have the same expected waiting time, and see that 12 is better for $p_1 > p$ and 121222211221 is better for $p_1 < p$. So, instead of using the original schedule 11212122221122, we should use either 12 or 121222211221, depending on the value of p_1.

3 Proof of Theorem 1

We begin the proof by classifying schedules into one of two sets. The first set is the "Reducible" schedules, the set of all schedules for which one of the lemmas applies to give a strictly better schedule. The second set is the "Irreducible" schedules, the set of all schedules for which no lemma can be used to give a strictly better schedule. We see that any schedule will be in exactly one of these two sets. We then look at the set of irreducible schedules, since any reducible schedule is worse than some irreducible schedule and hence not optimal. We compare these irreducible schedules and find that a small subset of them (the set of schedules listed in Theorem 1) forms the set of optimal schedules.

3.1 The Irreducible Schedules

Each of the lemmas provides a way to change a schedule to get another equally good or better schedule. We can think of the lemmas as describing actions we can perform on schedules to change them. There are two types of actions. For each type, we assume some fixed value of p_1.

The first type reduces the expected waiting time of a schedule. This type of action gives a schedule that is strictly better than the original. These actions establish a partial ordering, ρ_2, among schedules. When one schedule can be modified by one of these actions to get a second schedule, the second schedule is less than the first according to ρ_2.

The second type of action changes the structure of the schedule, but keeps the expected waiting time the same. This type of action does not give a measurably better schedule, but instead identifies schedules that are equal with respect to ρ_2. When one schedule can be modified by one of these actions to get a second schedule, the two schedules are equal.

There are two orderings of schedules. The first, ρ_1, is by EWT. Any two schedules can be compared using EWT. This is the ordering we use to define the optimal schedule at any value of p_1. The optimal schedule is simply the one that is "less than or equal to" all other schedules according to ρ_1.

The second ordering, ρ_2, is by the actions described above. Not all schedules can be compared by ρ_2, just the ones that are the initial and resulting schedules from some action. Any two that can be compared with ρ_2 will have the same ordering as by ρ_1, so ρ_2 is really a sub-ordering of ρ_1. That is, the set of relationships decribed by ρ_2 is a subset of the set of relationships described by ρ_1.

So, any minimal schedule under ρ_1 will be a minimal schedule under ρ_2. Our strategy for determining the optimal schedule (the minimal element under ρ_1) will be to determine the minimal elements under ρ_2 and then compare them under ρ_1 to find the optimal schedule.

For ease of referencing the lemmas and their associated transformations, we list the following actions that we can perform on schedules:

A1: $S = \mathbf{AB} \rightarrow S_1' = \mathbf{A}, S_2' = \mathbf{B}$, where both A and B start with the same subschedule C, which contains at least two 1's and two 2's.

A2: $S = \ldots 221\mathbf{21}\overbrace{(2)1(2)1\ldots(2)1}^{n\ 1's}22\ldots \rightarrow S' = \ldots 221\mathbf{12}\overbrace{(2)1(2)1\ldots(2)1}^{n\ 1's}22\ldots, n \geq 2$

A3: $S \rightarrow S' =$ reversal of S

A4: $S = [\ldots, a, \mathbf{n}, \mathbf{m}, b, \ldots] \rightarrow S' = [\ldots, a, \mathbf{2}, \mathbf{m{+}n{-}2}, b, \ldots], n \geq 2, m \geq 2, b \geq a$

A5: $S = [\ldots, a, \mathbf{b}, c, \ldots, d, \mathbf{e}, f, \ldots] \rightarrow S' = [\ldots, a, \mathbf{b{-}1}, c, \ldots, d, \mathbf{e{+}1}, f, \ldots], b \geq 3, e \geq 2$

A6: $S = \ldots 1\overbrace{22\ldots\mathbf{22}}^{n\ 2's}\overbrace{\mathbf{11}\ldots 11}^{m\ 1's}2\ldots \rightarrow S' = \ldots 1\overbrace{22\ldots 2}^{n-1}\mathbf{12}\overbrace{11\ldots 1}^{m-1}2\ldots, n \geq 3, m \geq 3$

We use the notation "N" to mean "three or more". Different occurances of N within a schedule can correspond to different numbers greater than or equal to three. For example, [0,N,1,2,N] can represent the schedule [0,3,1,2,3], [0,3,1,2,4], or [0,10,1,2,12], but not [0,3,1,2,1], [0,0,1,2,2], or [0,10,1,2,2].

We first show a weaker version of Theorem 1:

Proposition 1 *All optimal schedules are equal to a schedule or the complement of a schedule in the following list:*

$[0,1]$, $[0,1,1]$, $[0,1,2]$, $[0,1,2,N]$, $[0,1,N]$, $[0,1,N,2]$, $[0,1,N,2,N]$, $[0,2]$, $[0,2,1,N]$, $[0,2,N]$, $[0,2,N,1,N]$, $[0,N]$, $[0,N,1,2,N]$, $[0,N,1,N]$, $[0,N,1,N,2,N]$ $[0,N,2,N]$, $[1]$, $[1,2]$, $[1,2,N]$, $[1,N]$, $[1,N,2,N]$, $[2]$, $[2,N]$.

Note that the optimal schedules [0,2], [0,3], [0,4], [2], and [2,N] are all included in the list. We show later that these are the best schedules in the list. We now derive this list containing all the irreducible schedules.

First we consider all schedules that don't contain both three consecutive 1's and three consecutive 2's. For now, we assume there are no 111's. We will eliminate certain schedules based on the fact that we can find a better schedule according to ρ_2. Since we can find a better schedule, the original schedule is reducible, and we can exclude it from the list.

We know that no irreducible schedule can contain more than one each of 1122, 2211, 1212, 2121, 1221, or 2112, since we could use action A1 on such a schedule to get a better schedule. Here, and in general, we ignore schedules that are optimal at exactly one point, such as the schedule before splitting, at the value of p_1 where the two subschedules have the same waiting time as the original. This is because the other two schedules are not only optimal at that point, but also at neighboring points.

We can not have a schedule with two 0's in it, since we would have [... , 0, ... , 0, ...], which gives us two 2112's, or [... , 0, 0, ...], which gives us a 111. We can not have a schedule with two 2's in it, since we would have [... , 2, ... , 2, ...], which gives us two 1221's. If we have a schedule with two 1's in it, we have two instances of 121. If both are preceded by or followed by something other than 0, we get two instances of 1212 or 2121. The only way to prevent this is for one to be preceded by 0 and the other to be followed by 0. Since we can have at most one 0, we need to have [... , 1, 0, 1, ...], with only 2's or larger in the rest of the schedule. But with 2's in the schedule, we can use action A2 to get a better schedule. So, the only irreducible schedule with two 1's is [0, 1, 1].

We can not have a schedule with two adjacent N's, where N represents some number greater than 2, since we can use action A4 on such a schedule to get a better schedule. So, the irreducible schdules are [0,1,1] and all schedules that have at most one 0, 1, and 2, and no adjacent N's. It is straight-forward to list these:

[0,1,1], [N], [0], [0,N], [1], [1,N], [2], [2,N], [0,1], [0,1,N], [0,N,1], [0,N,1,N], [0,2], [0,2,N], [0,N,2], [0,N,2,N], [1,2], [1,2,N], [1,N,2], [1,N,2,N], [0,1,2], [0,1,2,N], [0,1,N,2], [0,N,1,2], [0,1,N,2,N], [0,N,1,2,N], [0,N,1,N,2], [0,2,1], [0,2,1,N], [0,2,N,1], [0,N,2,1], [0,2,N,1,N], [0,N,2,1,N], [0,N,2,N,1], [1,0,2], [1,0,2,N], [1,0,N,2], [1,N,0,2], [1,0,N,2,N], [1,N,0,2,N], [1,N,0,N,2], [1,2,0], [1,2,0,N], [1,2,N,0], [1,N,2,0], [1,2,N,0,N], [1,N,2,0,N], [1,N,2,N,0], [2,0,1], [2,0,1,N], [2,0,N,1], [2,N,0,1], [2,0,N,1,N], [2,N,0,1,N], [2,N,0,N,1], [2,1,0], [2,1,0,N], [2,1,N,0], [2,N,1,0], [2,1,N,0,N], [2,N,1,0,N], [2,N,1,N,0], [0,N,1,N,2,N], [0,N,2,N,1,N].

We can eliminate repetitions of the same schedule using periodicity of the schedules. For example, [0,2,N,1] = [1,0,2,N]. We can use action A3 to eliminate reversals. We also eliminate [0], since it does not contain item 2. When we do this, we get the following list of schedules:

[0,1], [0,1,1], [0,1,2], [0,1,2,N], [0,1,N], [0,1,N,2], [0,1,N,2,N], [0,2], [0,2,1,N], [0,2,N], [0,2,N,1,N], [0,N], [0,N,1,2,N], [0,N,1,N], [0,N,1,N,2,N] [0,N,2,N], [1], [1,2], [1,2,N], [1,N], [1,N,2,N], [2], [2,N].

If we allow 111 and not 222, we get the complements of the schedules in this list. Thus, the set of these schedules and their complements contains all irreducible schedules that do not have both a sequence of 111 and a sequence of 222.

We now consider schedules with both 111 and 222 sequences. We can decompose any such schedule into a sequence of subschedules that begin and end with either 111 or 222, and have no sequences of three or more 1's or 2's other than at their beginning or end. We will now derive the set of irreducible subschedules that can be combined to form irreducible schedules.

To do this, we start with the sequence 1112 and extend this schedule one piece at a time until we reach either a 111...111 subschedule, a 111...222 subschedule, or a reducible subschedule (a subschedule such that any schedule containing it is reducible). For each position, we can choose either 1 or 2, so we search all possibilities using a binary tree. This is diagrammed in figure 2.

From this tree, we see that there are seven irreducible 111...222 subschedules. We label them A through G. The irreducible 222...111 subschedules are simply the complements of the irreducible 111...222 subschedules. We will call these $\overline{\text{A}}$ through $\overline{\text{G}}$. We now generate 111...222... schedules by combining these 111...222 and 222...111 subschedules. We use action A1 to eliminate reducible schedules and we find that the only irreducible combinations are C$\overline{\text{C}}$, C$\overline{\text{E}}$, E$\overline{\text{C}}$, F$\overline{\text{G}}$, G$\overline{\text{F}}$, and G$\overline{\text{G}}$.

Of these, only C$\overline{\text{C}}$ and G$\overline{\text{G}}$ do not contain all six possible patterns of length four that have two

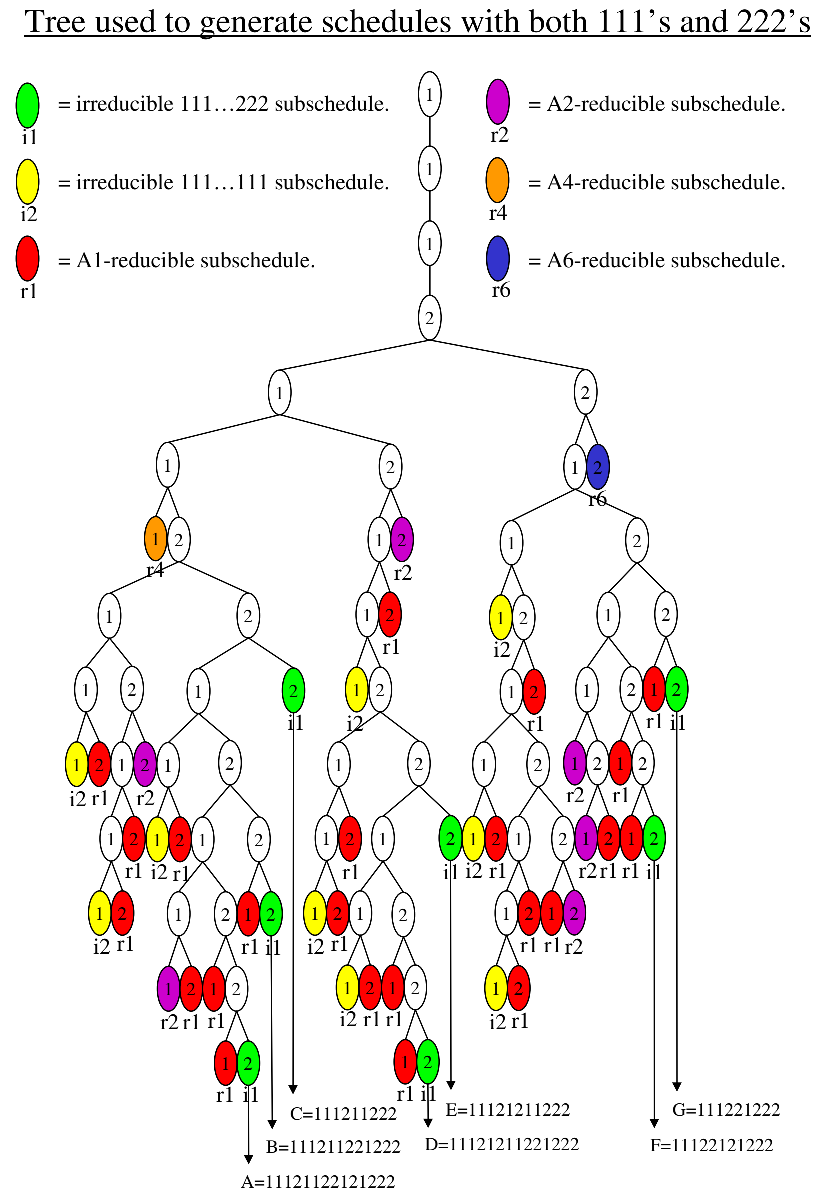

Figure 2: Tree of schedules

each of item 1 and item 2. In these we can replace 111 with 111...111, if the 111...111 subschedule does not have any of the same patterns in it as the starting schedule. We find that the only such 111...111 subschedule is 111212111. When this subschedule is added to $C\overline{C}$ and $G\overline{G}$, each resulting schedule contains all six patterns. So, the only possible schedules with both 111 and 222 are: $C\overline{C}$, $C\overline{E}$, $E\overline{C}$, $F\overline{G}$, $G\overline{F}$, $G\overline{G}$, $C'\overline{C}$, $G'\overline{G}$, $C\overline{C'}$, and $G\overline{G'}$, where C' is C with 111 replaced by 111212111 and G' is G with 111 replaced by 111212111.

We need only consider $C\overline{C}$, $C\overline{E}$, and $C'\overline{C}$, since $E\overline{C} = (C\overline{E})^C$, $F\overline{G} = (C\overline{E})^R$, $G\overline{F} = (C\overline{E})^{CR}$, $G\overline{G} = (C\overline{C})^{CR}$, $G'\overline{G} = (C'\overline{C})^R$, $C\overline{C'} = (C'\overline{C})^C$, and $G\overline{G'} = (C'\overline{C})^{CR}$. We use action A4 on $C\overline{C}$ = 111211222122 to get 111211221222. We then use action A6 to get the schedule 211211221221, to which we can apply action A2. We can do the same thing to $C\overline{E}$ = 11121122212122 and $C'\overline{C}$ = 111212111211222122, applying action A4 to give 11211122212122 and 111212112111222122, respectively, and then A6 to get 11211212212122 and 111212112112122122, which we can reduce using action A2.

Since all of the schedules listed above can be reduced, there are no irreducible schedules with both a 111 and a 222. So, the schedules listed previously and their complements are the only possible schedules that can not be reduced to better schedules, and Proposition 1 follows.

3.2 The Optimal Schedules

We have thus far shown that the schedules 1122, 11222, 112222, and $1221\overbrace{22\ldots22}^{n\geq 2}$ are irreducible. Now we finish the proof by showing they are better than the other irreducible schedules:

Proposition 2 *At any value of p_1, EWT(S, p_1) is minimized over all irreducible S by one of* 1122, 11222, 112222, *or* $1221\overbrace{22\ldots22}^{n\geq 2}$.

To show this, we first compute their expected waiting times as a function of p_1. We then plot expected waiting time vs. p_1 for each schedule. We find the appropriate intersection points and see that 1122 is best on $p_1 \in \left(\frac{5}{16}, \frac{1}{2}\right)$, 11222 is best on $\left(\frac{5}{21}, \frac{5}{16}\right)$, 112222 is best on $\left(\frac{1}{5}, \frac{5}{21}\right)$, 122122 is best on $\left(\frac{2}{17}, \frac{1}{5}\right)$, and $12212\overbrace{22\ldots22}^{n\geq 1}$ is best on $\left(\frac{8}{(n+1)^2+13(n+1)+68}, \frac{8}{n^2+13n+68}\right)$. This is illustrated in Figure 3.

It remains to show that any other irreducible schedule is worse than one of these schedules for any value of p_1. Suppose there is some schedule, with graph defined by $t = a \cdot p_1 + b$, that has a shorter expected waiting time at some p_1 than any of our optimal schedules. Then it will dip below the (piecewise linear) minimum function of the graphs. Since this function is linear, it must be less at some corner of the minimum function. So, if we can show that no schedule has a graph that dips below a corner, we have shown that our schedules are optimal.

We do this by checking the set $C_1 = \left\{\left(\frac{5}{16}, \frac{7}{4}\right), \left(\frac{5}{21}, \frac{71}{42}\right), \left(\frac{1}{5}, \frac{49}{30}\right)\right\}$ of intersection points of 1122 & 11222, 11222 & 112222, and 112222 & 122, and the set $C_2 = \left\{\left(\frac{8}{n^2+13n+68}, \frac{1}{2} + \frac{8(n^2+14n+48)}{n^3+19n^2+146n+408}\right) \middle| n \geq 0\right\}$, where $122122\overbrace{2\ldots2}^{n}$ and $122122\overbrace{2\ldots2}^{n+1}$ intersect, $\forall n \geq 0$. These are our corner points. We evaluate $t = a \cdot p_1 + b$ at the corner point (p_c, t_c) and compute the difference $t - t_c = a \cdot p_c + b - t_c$. We then show this is positive at all corner points.

As an example, consider the schedule $[0,N]$. If $m = N - 2$, this has expected waiting time $\frac{m^2+7m+10}{4(m+4)}$ for item 1 and $\frac{5}{4(m+4)}$ for item 2. So, $t - t_c$ at the "n^{th}" corner point in C_2 is

$$\frac{m^2+7m}{4m+16} \cdot \frac{8}{n^2+13n+68} + \frac{5}{2m+8} - \frac{4(n^2+14n+48)}{n^3+19n^2+146n+408}$$

This is non-negative $\Leftrightarrow$

$$4(m^2+7m)(n+6) + 5(n^3+19n^2+146n+408) - 4(2m+8)(n^2+14n+48) \geq 0$$

We rewrite as a quadratic in m to get

$$(4n+24)m^2 - (8n^2+84n+216)m + (5n^3+63n^2+282n+504) \geq 0$$

We know this is always positive if it has no real roots, so it is always positive if its discriminant is negative. So, we want

$$(8n^2+84n+216)^2 - 4(4n+24)(5n^3+63n^2+282n+504) < 0$$

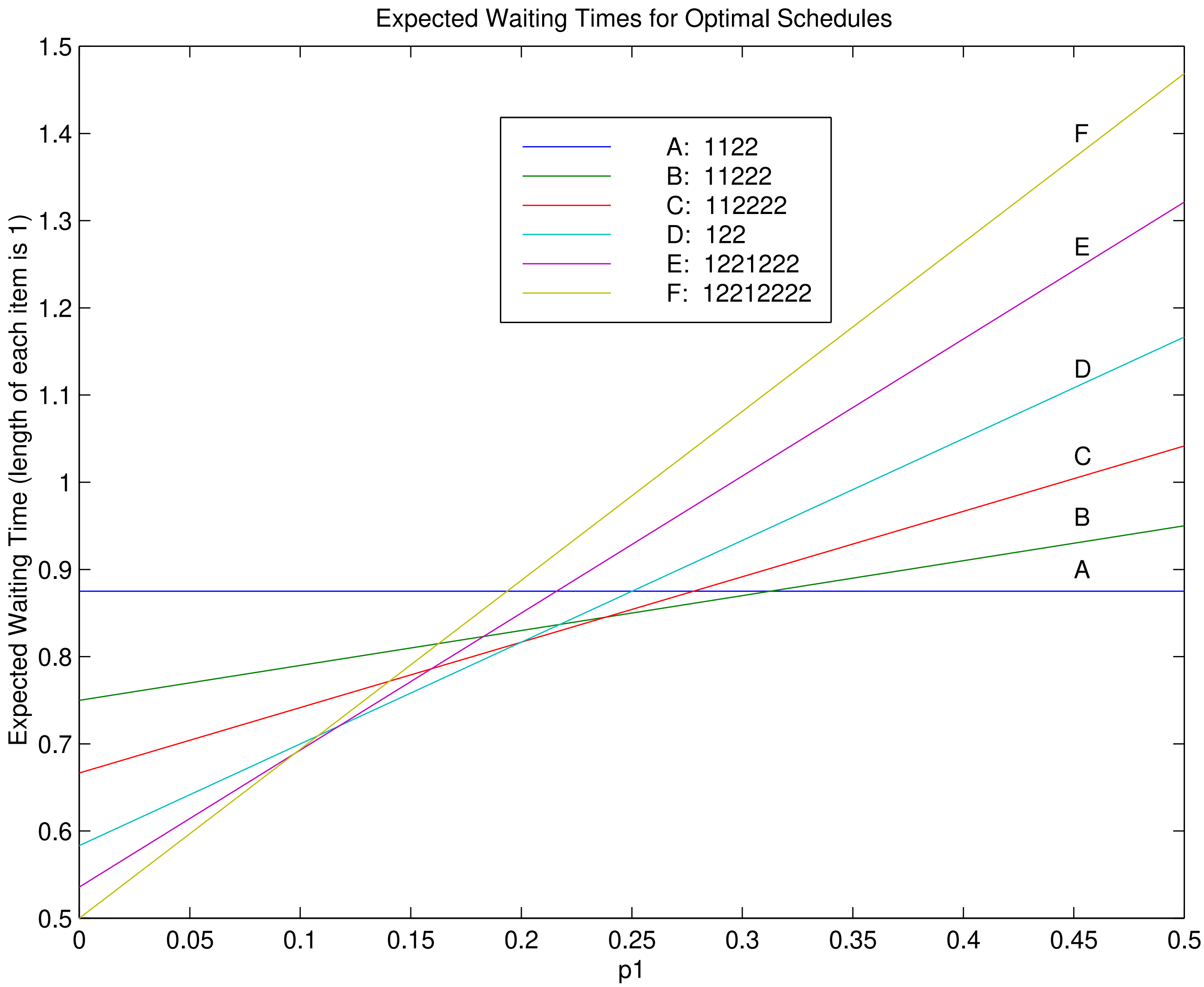

Figure 3: Expected Waiting time vs. p_1 for some of the optimal schedules

Simplifying, we get

$$16n^4 + 144n^3 + 48n^2 - 1152n + 1728 > 0$$

This is true for $n = 0, 1, 2$. For $n \geq 3$, note that this expression is greater than $16(n-3)^4$, which is non-negative for all $n \geq 3$. So, this expression is always positive, and hence there are no values of $m \geq 0$ and $n \geq 0$ where the schedule $[0,N]$ is better than $122122\overbrace{2\ldots2}^{n}$. We also check against 1122, 11222, 112222, and 122122, and see that we do not dip below their corner points. We do this the same way, except now we only have the single parameter n instead of both m and n.

We use the same basic idea for all other irreducible schedules. Some of them have more than one group of N 2's. For these, we use action A5 to determine how many 2's the blocks can have relative to each other, and we consider all possible combinations, doing the above calculation for each. So, Proposition 2 is proved, and combining with Proposition 1 we see that our small set of schedules $\{1122, 11222, 112222, 1221\overbrace{22\ldots22}^{n\geq 2}\}$ performs better than any other schedule. The intervals on which each is optimal are described by C_1 and C_2. These agree with the intervals in

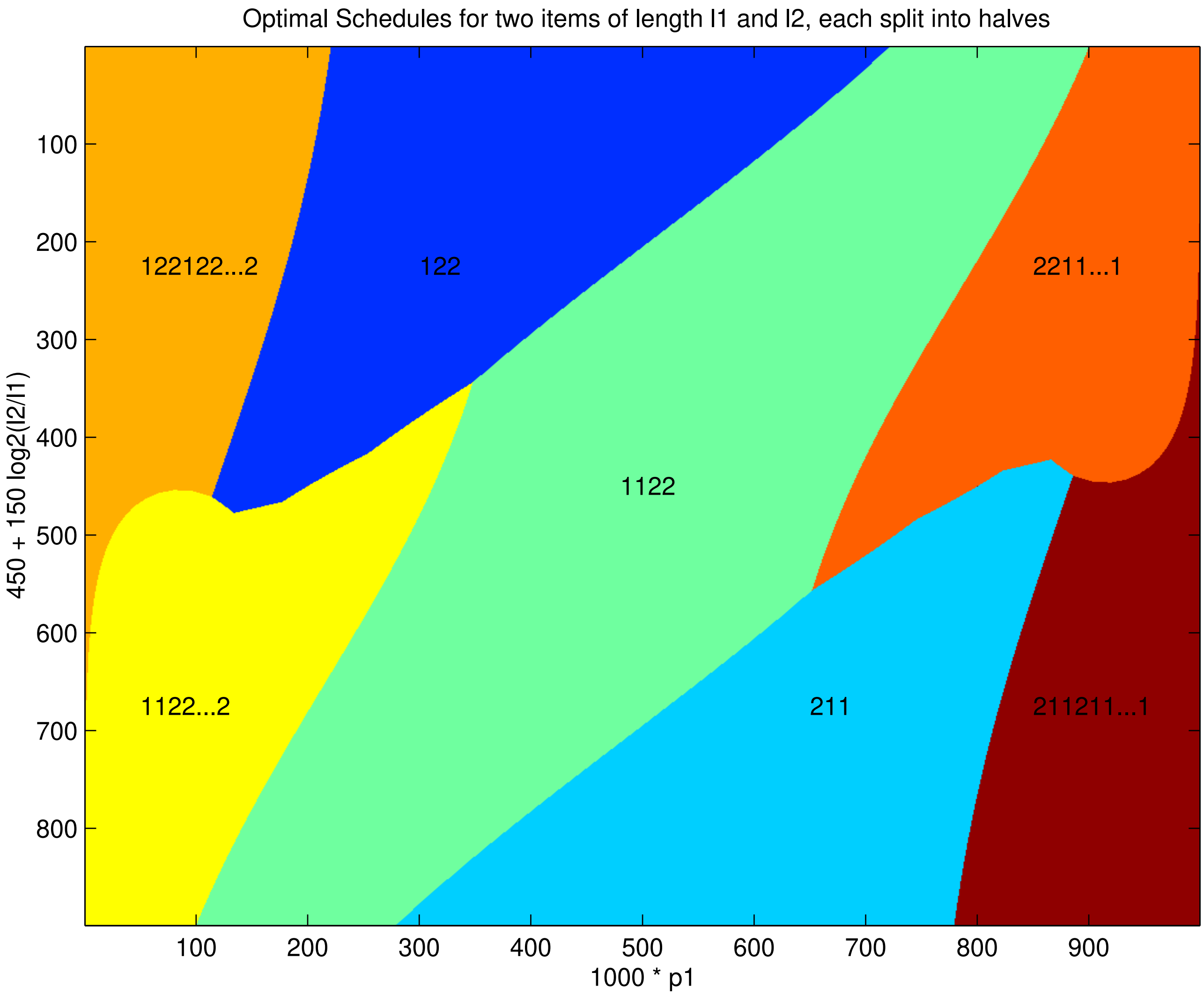

Figure 4: Optimal scheduling with different lengths

the statement of Theorem 1, thus completing the proof.

4 Different Length Items

We have shown the optimal scheduling for two items of the same length, when we split them each into halves. Now we consider the same situation, but with items of different lengths. We fix the length of item 1 at $l_1 = 1$ and let item 2 have length l_2 that can be any positive value. We then split item 1 into two pieces, each of size $\frac{1}{2}$, and split item 2 into two pieces, each of size $\frac{l_2}{2}$. We want to find the optimal schedule for the items, as a function of the ratio of the lengths, l_2, and the demand probabilities p_1 and $p_2 = 1 - p_1$.

We attempt to use the same reasoning as with $l_2 = l_1$. Since the lemmas we used for equal lengths also hold true for different lengths, we can use the same reductions and arrive at the same set of irreducible schedules. However, comparing the schedules to each other is more difficult than when l_2 is fixed at 1.

To get an idea about which schedules are optimal for which values of l_2 and p_1, we numerically checked a range of values of l_2 and p_1, and found the optimal schedule at each pair of values. The results are shown in figure 4. We see that the only optimal schedules are 1122...2, 122122...2, and their complements. It is interesting to note that these are the same optimal schedules as for equal length items.

It would be interesting to know if this is provably true for all p_1 and l_2. We have been able to eliminate some of the schedules as being non-optimal. For example, [1,n+2] is always worse than [0,n+3], for all l_2, p_1, and $n \geq 0$, so [1,N] is worse than [0,N] and hence [1,N] is never an optimal schedule. However, we have not been able to eliminate all other schedules, as we could for $l_2 = l_1$.

5 Future Work

Future work involves finding the best schedules for more than two items, of varying lengths. We have shown that a single split can improve expected waiting time. More work can be done to see the effects of multiple splits on expected waiting time. We would also like to look at the effects of transmission errors and ways to effectively combat them, as well as adding multiple transmission channels to improve bandwidth.

References

[1] D. Aksoy, M. J. Franklin, “Scheduling for Large-Scale On-Demand Data Broadcasting”, IEEE INFOCOM, San Francisco, CA, March 1998.

[2] A. Bar-Noy, R. Bhatia, J. Naor, B. Schieber, “Minimizing Service and Operations Cost of Periodic Scheduling”, 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 1998.

[3] A. Bestavros, “AIDA-based Real-Time Fault-Tolerant Broadcast Disks”, Technical Report 96-001, Computer Science, Boston University, January 5, 1996.

[4] M. J. Franklin, S. Zdonik, “Dissemination-Based Information Systems”, IEEE Data Engineering Bulletin, Vol 19, No 3, September 1996.

[5] S. Hameed, *Scheduling Information Broadcast in Asymmetric Environment*, Masters of Science thesis, Texas A&M University, May 1997.

[6] S. Hameed, N. H. Vaidya, “Log-time Algorithms for Scheduling Single and Multiple Channel Data Broadcast”, MOBICOM’97, Budapest, September 1997.

[7] S. Jiang, N. H. Vaidya, “Scheduling Algorithms for a Data Broadcast System: Minimizing Variance of the Response Time”, Technical Report 98-005, Computer Science, Texas A&M University, February 4, 1998.

[8] H. V. Leong, A. Si, “Database Caching Over the Air-Storage”, *The Computer Journal* 40(7): 401-415, 1997.

[9] N. H. Vaidya, S. Hameed, “Data Broadcast in Asymmetric Wireless Environments”, First International Workshop on Satellite-based Information Services (WOSBIS), Rye, NY, November 1996.

[10] S. Zdonik, M. J. Franklin, R. Alonso, S. Acharya, “Are ’Disks in the Air’ Just ’Pie in the Sky’?”, IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, December 1994.

A Appendix: Proof of the Lemmas

A.1 Proof of Lemma 1

A.1.1 Lemma 1 Part (a)

$S = \cdots 12^a 12^b 12^c 1 \cdots 12^d 12^e 12^f 1 \cdots, S' = \cdots 12^a 12^{b-1} 12^c 1 \cdots 12^d 12^{e+1} 12^f 1 \cdots$, $b \geq 3, e \geq 2$.

S and S' have the same length, l_S. We fix $l_1 = 1$ and let l_2 be any positive value. To show $EWT(S', p_1) \leq EWT(S, p_1) \,\forall p_1$, we show $\Delta t = EWT(S, p_1) - EWT(S', p_1) \geq 0 \,\forall p_1$. We compute Δt as $p_1 \Delta t_1 + p_2 \Delta t_2 = p_1 \left(t_1 - t_1'\right) + p_2 \left(t_2 - t_2'\right)$, where t_1 and t_2 are the waiting times for items 1 and 2 using schedule S, and t_1' and t_2' are the waiting times for items 1 and 2 using schedule S'. The algebra to show this follows. In each expression, we have a sum of terms of the form $length(time_1 + ... + time_n)$. The $length$ value corresponds to arriving in a section of the schedule of length $length$. The sum of the times is the sum of the lengths of the nondesired sections that one waits through before getting all of the desired item, when arriving in the section of length $weight$. Adding all terms of this form and dividing by the total length of the schedule gives us the waiting time for an item.

$$
\begin{aligned}
t_1 &= \frac{1}{l_S}\left[\cdots + 1\left(\frac{1}{2} + al_2 + bl_2\right) + al_2\left(\frac{al_2}{2} + bl_2\right) + 1\left(\frac{1}{2} + bl_2 + cl_2\right) + bl_2\left(\frac{bl_2}{2} + cl_2\right) + \cdots \right.\\
&\quad \left. +1\left(\frac{1}{2} + dl_2 + el_2\right) + dl_2\left(\frac{dl_2}{2} + el_2\right) + 1\left(\frac{1}{2} + el_2 + fl_2\right) + el_2\left(\frac{el_2}{2} + fl_2\right) + \cdots\right]\\
t_1' &= \frac{1}{l_S}\left[\cdots + 1\left(\frac{1}{2} + al_2 + bl_2 - l_2\right) + al_2\left(\frac{al_2}{2} + bl_2 - l_2\right) + 1\left(\frac{1}{2} + bl_2 + cl_2 - l_2\right)\right.\\
&\quad +bl_2\left(\frac{bl_2}{2} - \frac{l_2}{2} + cl_2\right) - l_2\left(\frac{bl_2}{2} - \frac{l_2}{2} + cl_2\right) + \cdots + 1\left(\frac{1}{2} + dl_2 + el_2 + l_2\right)\\
&\quad +dl_2\left(\frac{dl_2}{2} + el_2 + l_2\right) + 1\left(\frac{1}{2} + el_2 + fl_2 + l_2\right) + el_2\left(\frac{el_2}{2} + \frac{l_2}{2} + fl_2\right)\\
&\quad \left. +l_2\left(\frac{el_2}{2} + \frac{l_2}{2} + fl_2\right) + \cdots\right]\\
\Delta t_1 &= t_1 - t_1'\\
&= \frac{1}{l_S}\left[1\left(l_2\right) + al_2\left(l_2\right) + 1\left(l_2\right) + bl_2\left(\frac{l_2}{2}\right) + l_2\left(\frac{bl_2}{2} - \frac{l_2}{2} + cl_2\right)\right.\\
&\quad \left. +1\left(-l_2\right) + dl_2\left(-l_2\right) + 1\left(-l_2\right) + el_2\left(-\frac{l_2}{2}\right) - l_2\left(\frac{el_2}{2} + \frac{l_2}{2} + fl_2\right)\right]\\
&= \frac{1}{l_S}\left[l_2^2\left(a + b + c - d - e - f - 1\right)\right]
\end{aligned}
$$

It is easy to see that $t_2 = t_2'$, so $\Delta t_2 = 0$. It follows that $\Delta t \geq 0 \iff (a+b+c) - (d+e+f) \geq 1$. Thus, $EWT(S', p_1) \leq EWT(S, p_1) \iff (a+b+c) - (d+e+f) \geq 1$, and part (a) of the lemma is proved.

A.1.2 Lemma 1 Part (b)

$T = \cdots 12^n 1212^x 12^y 1 \cdots, T' = \cdots 12^n 1122^x 12^y 1 \cdots, n \geq 2, x, y \in \{0, 1\}$.

T and T' have the same length, l_T. We fix $l_1 = 1$ and let l_2 be any positive value. To show $EWT(T', p_1) \leq EWT(T, p_1) \,\forall p_1$, we show $\Delta t = EWT(T, p_1) - EWT(T', p_1) \geq 0 \,\forall p_1$. We compute Δt as $p_1 \Delta t_1 + p_2 \Delta t_2 = p_1 (t_1 - t_1') + p_2 (t_2 - t_2')$, where t_1 and t_2 are the waiting times for items

1 and 2 using schedule T, and t_1' and t_2' are the waiting times for items 1 and 2 using schedule T'. The algebra to show this follows. In each expression, we have a sum of terms of the form $length(time_1 + ... + time_n)$. The $length$ value corresponds to arriving in a section of the schedule of length $length$. The sum of the times is the sum of the lengths of the nondesired sections that one waits through before getting all of the desired item, when arriving in the section of length $weight$. Adding all terms of this form and dividing by the total length of the schedule gives us the waiting time for an item.

$$
\begin{aligned}
t_1 &= \frac{1}{l_T}\left[\cdots + 1\left(\frac{1}{2} + nl_2 + l_2\right) + nl_2\left(\frac{nl_2}{2} + l_2\right) + 1\left(\frac{1}{2} + l_2 + xl_2\right)\right. \\
&\quad \left. + l_2\left(\frac{l_2}{2} + xl_2\right) + 1\left(\frac{1}{2} + xl_2 + yl_2\right) + \cdots\right] \\
t_1' &= \frac{1}{l_T}\left[\cdots + 1\left(\frac{1}{2} + nl_2\right) + nl_2\left(\frac{nl_2}{2}\right) + 1\left(\frac{1}{2} + l_2 + xl_2\right)\right. \\
&\quad \left. + 1\left(\frac{1}{2} + l_2 + xl_2 + yl_2\right) + l_2\left(\frac{l_2}{2} + xl_2 + yl_2\right) + \cdots\right] \\
\Delta t_1 &= t_1 - t_1' \\
&= \frac{1}{l_T}\left[1\left(l_2\right) + nl_2\left(l_2\right) + 1\left(0\right) + l_2\left(-yl_2\right) + 1\left(-l_2\right)\right] \\
&= \frac{1}{l_T}\left[\left(n - y\right) l_2^2\right] \\
t_2 &= \frac{1}{l_T}\left[\cdots + 1\left(\frac{1}{2}\right) + (n-2)\, l_2\left(\frac{l_2}{2}\right) + l_2\left(\frac{l_2}{2} + 1\right) + l_2\left(\frac{l_2}{2} + 1 + 1 + T_2\right) + 1\left(\frac{1}{2} + 1 + T_2\right)\right. \\
&\quad \left. + l_2\left(\frac{l_2}{2} + 1 + T_{22}\right) + 1\left(\frac{1}{2} + T_{22}\right) + \cdots\right] \\
t_2' &= \frac{1}{l_T}\left[\cdots + 1\left(\frac{1}{2}\right) + (n-2)\, l_2\left(\frac{l_2}{2}\right) + l_2\left(\frac{l_2}{2} + 1 + 1\right) + l_2\left(\frac{l_2}{2} + 1 + 1 + T_2\right)\right. \\
&\quad \left. + 1\left(\frac{1}{2} + 1 + T_2\right) + 1\left(\frac{1}{2} + T_2\right) + l_2\left(\frac{l_2}{2} + T_{22}\right) + \cdots\right] \\
\Delta t_2 &= t_2 - t_2' \\
&= \frac{1}{l_T}\left[1\left(0\right) + (n-2)\, l_2\left(0\right) + l_2\left(-1\right) + l_2\left(0\right) + 1\left(0\right) + l_2\left(1\right) + 1\left(T_{22} - T_2\right)\right] \\
&= \frac{1}{l_T}\left[T_{22} - T_2\right]
\end{aligned}
$$

Here T_2 is the time to get one piece of item 2 when we start listening at the beginning of "2^x" and T_{22} is the time to get two pieces of item 2 when we start listening at the beginning of "2^x". It is easy to see that $T_{22} \geq T_2$. Also, $n \geq y$, by the restrictions we placed on them. So, $\Delta t_1 \geq 0$ and $\Delta t_2 \geq 0$. Since $\Delta t = p_1 \Delta t_1 + p_2 \Delta t_2$, we see that $\Delta t \geq 0$, and part (b) is proved.

A.1.3 Lemma 1 Part (c)

$U = \cdots 21^r 2^s 1^t 2^u 1 \cdots, U' = 21^r 2^s 1^{t-1} 212^{u-1} 1 \cdots, r, s \geq 1, t, u \geq 3.$

U and U' have the same length, l_U. We fix $l_1 = 1$ and let l_2 be any positive value. To show $EWT(U', p_1) \leq EWT(U, p_1)\, \forall p_1$, we show $\Delta t = EWT(U, p_1) - EWT(U', p_1) \geq 0\, \forall p_1$. We compute Δt as $p_1 \Delta t_1 + p_2 \Delta t_2 = p_1(t_1 - t_1') + p_2(t_2 - t_2')$, where t_1 and t_2 are the waiting times for items 1 and 2 using schedule U, and t_1' and t_2' are the waiting times for items 1 and 2 using schedule

U'. The algebra to show this follows. In each expression, we have a sum of terms of the form $length(time_1 + ... + time_n)$. The $length$ value corresponds to arriving in a section of the schedule of length $length$. The sum of the times is the sum of the lengths of the nondesired sections that one waits through before getting all of the desired item, when arriving in the section of length $weight$. Adding all terms of this form and dividing by the total length of the schedule gives us the waiting time for an item.

$$
\begin{aligned}
t_1 &= \frac{1}{l_U}\Bigg[\cdots + l_2\left(\frac{l_2}{2} + (r=1)\, sl_2\right) + (r>2)\,(r-2)\left(\frac{1}{2}\right) + (r\geq 2)\left(\frac{1}{2} + sl_2\right) + \left(\frac{1}{2} + sl_2\right) \\
&\quad + (s>2)\,(s-2)\, l_2\left(\frac{(s-2)\, l_2}{2} + 2l_2\right) + (s\geq 2)\, l_2\left(\frac{l_2}{2} + l_2\right) + l_2\left(\frac{l_2}{2}\right) + (t-2)\left(\frac{1}{2}\right) \\
&\quad + 1\left(\frac{1}{2} + ul_2\right) + 1\left(\frac{1}{2} + ul_2 + T_1\right) + ul_2\left(\frac{ul_2}{2} + T_1\right) + \cdots\Bigg] \\
t_1' &= \frac{1}{l_U}\Bigg[\cdots + l_2\left(\frac{l_2}{2} + (r=1)\, sl_2\right) + (r>2)\,(r-2)\left(\frac{1}{2}\right) + (r\geq 2)\left(\frac{1}{2} + sl_2\right) + \left(\frac{1}{2} + sl_2\right) \\
&\quad + (s>2)\,(s-2)\, l_2\left(\frac{(s-2)\, l_2}{2} + 2l_2\right) + (s\geq 2)\, l_2\left(\frac{l_2}{2} + l_2\right) + l_2\left(\frac{l_2}{2}\right) + (t-3)\left(\frac{1}{2}\right) \\
&\quad + 1\left(\frac{1}{2} + l_2\right) + 1\left(\frac{1}{2} + ul_2\right) + l_2\left(\frac{l_2}{2} + ul_2 - l_2\right) + 1\left(\frac{1}{2} + ul_2 - l_2 + T_1\right) \\
&\quad + (u-1)\, l_2\left(\frac{(u-1)\, l_2}{2} + T_1\right) + \cdots\Bigg] \\
\Delta t_1 &= t_1 - t_1' \\
&= \frac{1}{l_U}\Bigg[l_2\,(0) + (r>2)\,(r-2)\,(0) + (r\geq 2)\,(0) + (0) + (0) + (0) + l_2\,(0) + \frac{1}{2} + (u-1)\, l_2 + T_1 \\
&\quad + ul_2\left(\frac{l_2}{2}\right) + l_2\left(\frac{(u-1)\, l_2}{2} + T_1\right) - l_2\left(\frac{l_2}{2} + ul_2 - l_2\right) - 1\left(\frac{1}{2} + ul_2 - l_2 + T_1\right)\Bigg] \\
&= \frac{1}{l_U}\left[l_2 T_1\right] \\
t_2 &= \frac{1}{l_U}\Bigg[\cdots + l_2\left(\frac{l_2}{2} + r + (s=1)\, t\right) + r\left(\frac{r}{2} + (s=1)\, t\right) + (s>2)\,(s-2)\, l_2\left(\frac{l_2}{2}\right) \\
&\quad + (s\geq 2)\, l_2\left(\frac{l_2}{2} + t\right) + l_2\left(\frac{l_2}{2} + t\right) + t\left(\frac{t}{2}\right) + l_2\left(\frac{l_2}{2}\right) + \cdots\Bigg] \\
t_2' &= \frac{1}{l_U}\Bigg[\cdots + l_2\left(\frac{l_2}{2} + r + (s=1)\,(t-1)\right) + r\left(\frac{r}{2} + (s=1)\,(t-1)\right) + (s>2)\,(s-2)\, l_2\left(\frac{l_2}{2}\right) \\
&\quad + (s\geq 2)\, l_2\left(\frac{l_2}{2} + t - 1\right) + l_2\left(\frac{l_2}{2} + t\right) + (t-1)\left(\frac{t-1}{2} + 1\right) + l_2\left(\frac{l_2}{2} + 1\right) + 1\left(\frac{1}{2}\right) + \cdots\Bigg] \\
\Delta t_2 &= t_2 - t_2' \\
&= \frac{1}{l_U}\left[l_2\left((s=1)\, 1\right) + r\left((s=1)\,(1)\right) + (s>2)\,(0) + (s\geq 2)\, l_2 + t - 2 - l_2\right] \\
&= \frac{1}{l_U}\left[(s\geq 2)\, l_2 + (s=1)\,(l_2 + r) + t - 2 - l_2\right] \\
&= \frac{1}{l_U}\left[(s=1)\, r + t - 2\right]
\end{aligned}
$$

Here T_1 is the time to get one piece of item 1 when we start listening at the end of the piece of item 1 just after "2^u" in schedule U. This is the same as the time when we start listening at the end of the piece of item 1 just after "2^{u-1}" in schedule U'. Expressions such as $(r = 1)$ and $(s > 2)$ are evaluated as 1 if the expression in the parentheses is true and 0 if it is false. This is just a shorthand way of considering multiple cases with one equation. Since $T_1 \geq 0$, $r \geq 1$, and $t \geq 3$, we see that $\Delta t_1 \geq 0$ and $\Delta t_2 \geq 0$. Since $\Delta t = p_1 \Delta t_1 + p_2 \Delta t_2$, we see that $\Delta t \geq 0$, and part (c) is proved.

A.1.4 Lemma 1 Part (d)

$V = 1^{a_1} 2^{b_1} 1^{a_2} 2^{b_2} \cdots 1^{a_{k-1}} 2^{b_{k-1}} 1^{a_k} 2^{b_k}$, $V' = 2^{b_k} 1^{a_k} 2^{b_{k-1}} 1^{a_{k-1}} \cdots 2^{b_2} 1^{a_2} 2^{b_1} 1^{a_1}$. Without loss of generality, we assume all a_i's are 1.

V and V' have the same length, l_V. We fix $l_1 = 1$ and let l_2 be any positive value. To show $EWT(V', p_1) = EWT(V, p_1)\, \forall p_1$, we show $\Delta t = EWT(V, p_1) - EWT(V', p_1) = 0\, \forall p_1$. We compute Δt as $p_1 \Delta t_1 + p_2 \Delta t_2 = p_1(t_1 - t_1') + p_2(t_2 - t_2')$, where t_1 and t_2 are the waiting times for items 1 and 2 using schedule V, and t_1' and t_2' are the waiting times for items 1 and 2 using schedule V'. The algebra to show this follows. In each expression, we have a sum of terms of the form $length(time_1 + ... + time_n)$. The $length$ value corresponds to arriving in a section of the schedule of length $length$. The sum of the times is the sum of the lengths of the nondesired sections that one waits through before getting all of the desired item, when arriving in the section of length $weight$. Adding all terms of this form and dividing by the total length of the schedule gives us the waiting time for an item.

$$
\begin{aligned}
t_1 &= \frac{1}{l_V} \sum_{i=1}^{k} \left[1 \left(\frac{1}{2} + b_i l_2 + b_{(i+1) mod\, k} l_2 \right) + b_i l_2 \left(\frac{b_i l_2}{2} + b_{(i+1) mod\, k} l_2 \right) \right] \\
t_1' &= \frac{1}{l_V} \sum_{i=1}^{k} \left[b_i l_2 \left(\frac{b_i l_2}{2} + b_{(i-1) mod\, k} l_2 \right) + 1 \left(\frac{1}{2} + b_{(i-1) mod\, k} l_2 + b_{(i-2) mod\, k} l_2 \right) \right] \\
\Delta t_1 &= t_1 - t_1' \\
&= \frac{1}{l_V} \sum_{i=1}^{k} \left[l_2 \left(b_{(i+1) mod\, k} + b_i - b_{(i-1) mod\, k} - b_{(i-2) mod\, k} \right) \right. \\
&\quad \left. + b_i l_2 \left(b_{(i+1) mod\, k} l_2 - b_{(i-1) mod\, k} l_2 \right) \right] \\
&= \frac{1}{l_V} \left[l_2 \left(\sum_{i=1}^{k} b_{(i+1) mod\, k} + \sum_{i=1}^{k} b_i - \sum_{i=1}^{k} b_{(i-1) mod\, k} - \sum_{i=1}^{k} b_{(i-2) mod\, k} \right) \right. \\
&\quad \left. + b_i l_2^2 \left(\sum_{i=1}^{k} b_{(i+1) mod\, k} - \sum_{i=1}^{k} b_{(i-1) mod\, k} \right) \right] \\
&= \frac{1}{l_V} \left[l_2 \left(\sum_{i=1}^{k} b_i + \sum_{i=1}^{k} b_i - \sum_{i=1}^{k} b_i - \sum_{i=1}^{k} b_i \right) + b_i l_2^2 \left(\sum_{i=1}^{k} b_i - \sum_{i=1}^{k} b_i \right) \right] \\
&= 0
\end{aligned}
$$

Similarly, $\Delta t_2 = 0$. So, we have $\Delta t = 0$ and part (d) is proved.

A.2 Proof of Lemma 2

We use the result of lemma 1(a), plus an additional little trick. We write S as $SSSSSS$, repeating the schedule six times. This is still the same schedule, since we broadcast schedules repeatedly.

Now, we choose the b-length section from the second S and the e-length section from the fifth S, and choose a and c adjacent to b, and d and f adjacent to e in the overall schedule. There can be no overlap of these regions. We now apply lemma 1(a) to $SSSSSS$ to get SS^-SSS^+S. Here S^- is S with b decreased by one and S^+ is S with e increased by one. We will write $S^\pm$ to represent S with b decreased by one and e increased by one. We then do a cyclic shift and repeat on S^-SSS^+SS to get $S^-S^-SS^+S^+S$. We shift and repeat four more times, giving $S^\pm S^\pm S^\pm S^\pm S^\pm S^\pm = S^\pm = S'$ in lemma 1(a). At each step, we reduced EWT, so $EWT(S', p_1) \leq EWT(S, p_1)\, \forall p_1$, and lemma 2 is proved.

A.3 Proof of Lemma 3

We can write schedule S as

$$(*)\ \overbrace{\overbrace{|C_{1(start)} \cdots\cdots}^{S_1}\overbrace{|C_{2(start)} \cdots\cdots}^{S_2}}^{S}\overbrace{|C_{1(start)} \cdots\cdots}^{S_1}$$

where $C_{1(start)}$ indicates the start of the first instance of subschedule C within S and $C_{2(start)}$ indicates the start of the second occurance. Since S repeats cyclically, we can assume without loss of generality that it starts with one of the subschedules C.

We can also write schedules S_1 and S_2 as follows:

$$(**)\ \overbrace{|C_{(start)} \cdots\cdots}^{S_1}\overbrace{|C_{(start)} \cdots\cdots}^{S_1}$$

$$(***)\ \overbrace{|C_{(start)} \cdots\cdots}^{S_2}\overbrace{|C_{(start)} \cdots\cdots}^{S_2}$$

If we start waiting at some time within the first S_1 group in either (*) or (**), we will wait through a certain number of blocks before finding enough blocks of the desired item. If we stay within the initial S_1 group, the times are identical for (*) and (**) since the sequence of blocks that we encounter is the same in each case. If we must proceed into the next group, we first enter the C subschedule in either case. Since C contains two blocks each of items 1 and 2, we will not have to advance past the C group, so the waiting times in these cases are identical, since we again encounter the same sequence of blocks. So, it follows that the waiting time for schedule S, given that we arrive somewhere within subschedule S_1, is the same as it is for S_1 treated as a full schedule. The same applies for subschedule S_2. So, we have $EWT(S, p_1) = Pr(arrive\ in\ S_1) \cdot EWT(S_1, p_1) + Pr(arrive\ in\ S_2) \cdot EWT(S_2, p_1)$. The probabilities of arriving in subschedules S_1 and S_2 within schedule S are simply $\frac{l_1}{l}$ and $\frac{l_2}{l}$, respectively. So, we have $EWT(S, p_1) = \frac{l_1}{l} \cdot EWT(S_1, p_1) + \frac{l_2}{l} \cdot EWT(S_2, p_1)$.