Scheduling Data Broadcast to "Impatient" Users

Shu Jiang * Nitin H. Vaidya
Department of Computer Science
Texas A&M University
College Station, TX 77843-3112, USA
Email: {jiangs,vaidya}@cs.tamu.edu
Phone: 409-862-2598 Fax: 409-847-8578

Abstract

Broadcasting is an effective way of delivering data to a large population. In the broadcast environment under consideration, a server broadcasts data items to all clients simultaneously, according to a certain transmission schedule. Users with pending data requests need to listen to the broadcast channel until their requests are satisfied by the transmitted data. Past research on broadcast scheduling assumes that once a user starts to wait for some data item, the user waits until the desired data item is transmitted by the server. This is often not true in practice. For various reasons, users may loose patience after waiting "too long" and leave with their requests unserved.

In this paper, we study the broadcast scheduling problem taking user impatience into account. Based on our analytical results, we propose a scheduling algorithm that can produce a broadcast schedule with high service ratio (i.e., percentage of requests served) as well as low mean waiting time for the requests. Performance evaluation results based on simulations are provided.

1 Introduction

In a wireless environment, providing a large population of clients with access to data is a significant challenge [3]. In recent years, *data broadcasting* has been studied extensively (e.g., [5, 6, 9, 10, 17, 11, 12, 7]) as a mechanism for information delivery. In this approach, a server broadcasts the data over a broadcast channel which is listened to by all the users. All pending requests for a given data item are served when that item is transmitted on the broadcast channel. To avoid very long waiting time for the users, the broadcast schedule must be chosen by taking into account the demands for various data items.

Prior work on broadcast scheduling has considered different ways of reducing the waiting time and energy consumption, for instance, by scheduling the broadcast properly [5, 4, 16, 15, 3, 14], by utilizing client-side caching and prefetching [17, 2, 3, 1] or by energy-efficient indexing [11].

Prior research assumes that once a data request is generated by a user, the request will be held until it is satisfied. This assumption is not always true in practice. For various reasons, users may leave if their requests are still unsatisfied after waiting for some time. We refer to such users as "impatient" users.

*contact author

This paper provides a formalization of the problem of broadcast scheduling with impatient users, and proposes an algorithm to schedule broadcasts to such users. The objective of the design of a broadcast schedule is two-fold in this case: maximize the service ratio (i.e., percentage of the requests served), and minimize the mean waiting time.

The rest of this paper is organized as follows. Section 2 derives mathematical expressions for service ratio and mean waiting time. Based on these expressions, we derive an optimality condition for a broadcast schedule. This result is then used to propose a scheduling algorithm aimed at producing a near-optimal schedule with high service ratio as well as low mean waiting time. Section 3 presents performance measurements for the new algorithm. Section 4 presents our conclusions.

2 Mathematical Foundation and Proposed Algorithm

In this paper, we consider the *pure push* model wherein the broadcast schedule by the server is a function of *request probability distribution* for the data items in the server's database – the request probability distribution provides the server with a measure of the popularity of various data items. The actual requests pending at a given time are not known to the server.

Our results can be easily extended to the *pure pull* model wherein the broadcast schedule is a function of the requests known to be pending at a given time (thus, in the *pure pull* model, the server is aware of all the pending requests). For lack of space, we do not consider the *pure pull* model in this paper.

We now introduce some notations. The server maintains a database consisting of M data items numbered from 1 through M. Request probability p_i of data item i is a measure of the popularity of item i. Thus, p_i is the probability that item i is requested in any given request from a user. Length of item i, denoted as l_i, is the time required to broadcast item i.

Request Arrival Model

Similar to some past work on broadcast scheduling, users are assumed to submit independent requests according to a Poisson process with rate λ [6, 15]. Thus, requests for item i are generated according to a Poisson process with rate $\lambda_i = p_i\lambda$. Note that, when different requests are correlated, or when a user may make multiple requests at the same time, the analysis presented below will not apply.

User Impatience Model

As we mentioned before, a waiting user may leave before the requested data item is transmitted (thus, the user's request may be unsatisfied). In the following, we say that a request "arrives" when the request is generated by a user. Also, we say that a request "leaves" when a user withdraws (i.e., drops) the request before the request is served.

To facilitate the analysis, we need to characterize the duration of time a request waits before leaving. Let a random variable w represent the length of time that a request will wait before leaving. In this paper, we assume that w has an exponential distribution with mean $1/\tau$. Thus, τ can be regarded as the *rate* at which requests leave.

Service Ratio

Service ratio is the fraction of requests that are serviced before they leave. An objective of the proposed approach is to maximize the service ratio. We now derive an expression for the *expected service ratio*, denoted as R. We consider a broadcast schedule with the "equal spacing" property, i.e., in the broadcast schedule, consecutive transmissions of a given item are always separated by a fixed interval (or spacing).[1]

For the analysis below, we characterize a broadcast schedule satisfying the equal spacing property with a *schedule vector* $< s_1, s_2, \ldots, s_M >$, where s_i is the spacing between consecutive transmissions of item i.

A request, for item i, arriving at time t, will be satisfied by the next transmission of item i, provided that the request does not leave before that transmission begins. Thus, the expected number of requests satisfied by a transmission of item i, denoted as N_{s_i}, can be obtained as follows:

$$N_{s_i} = \int_0^{s_i} \lambda_i e^{-\tau(s_i - t)} dt = \frac{\lambda_i}{\tau}(1 - e^{-\tau s_i}) = \frac{p_i \lambda}{\tau}(1 - e^{-\tau s_i})$$

Note that the above expression calculates the number of those requests which arrive in the s_i interval preceding a given transmission of item i, but do not leave until the given transmission begins. Due to the exponential model, the probability that a request will wait for interval $(s_i - t)$ or longer is $e^{-\tau(s_i - t)}$.

If the length of entire broadcast schedule is C, then the number of broadcasts of item i is $n_i = C/s_i$, and the expected number of requests served by the schedule would be $\sum_{i=1}^{M} n_i N_{s_i}$. Also, the expected number of requests arriving in time period C is λC for a Poisson process with rate λ. Then, we obtain the expected service ratio R as

$$R = \frac{\sum_{i=1}^{M} n_i N_{s_i}}{\lambda C} = \sum_{i=1}^{M} \frac{p_i}{\tau s_i}(1 - e^{-\tau s_i}) \tag{1}$$

[1] It can be shown that, the theoretically optimal schedule satisfies the equal spacing property.

Mean Waiting Time

If a request is served (by a transmission from the server) after waiting for t time units, or if it leaves without being served after waiting for t time units, then its waiting time is said to be t. An analysis similar to that for the expected service ratio R yields the following expression for the mean waiting time of all requests (satisfied and unsatisfied both). The mean waiting time is denoted as W.

$$\begin{aligned} W &= \sum_{i=1}^{M} \frac{p_i}{\tau^2 s_i}(\tau s_i - 1 + e^{-\tau s_i}) \qquad (2) \\ &= \frac{1}{\tau}(1 - R) \qquad (3) \end{aligned}$$

Equations 1 and 3 imply that when the expected service ratio is maximized by a particular schedule vector, the expected mean waiting time will be minimized simultaneously.

Characteristics of the Optimal Schedule

From Equations 1 and 2, we know that service ratio R and mean waiting time W are multivariable functions of s_1, s_2, ..., s_M. Theoretically, we can use mathematical methods to locate the optimal point where R is maximized and W is minimized. The values of s_i's at the optimal point would correspond to the schedule that achieves the maximum service ratio and minimum mean waiting time. In the appendix, we present a derivation of a condition for optimality of a broadcast schedule. Result of this analysis is summarized in the following theorem.

Theorem 1 *Maximum service ratio and minimum mean waiting time are achieved if the broadcast schedule possesses the following property, for some constant K.*

$$\frac{p_i}{l_i}(\tau s_i e^{-\tau s_i} + e^{-\tau s_i} - 1) = K, \quad i = 1, 2, \ldots, M$$

Unfortunately, it is hard to obtain a closed-form solution for s_i's that satisfy the above equality for all i (under the constraint that the M items use the available broadcast bandwidth). However, Theorem 1 manifests an important relationship between item spacings in an optimal schedule, which provides a basis for the design of the proposed algorithm (we believe that our algorithm can perform near-optimally).

Broadcast scheduling scheme

The proposed broadcast scheduling algorithm is an adaptation of an algorithm we previously introduced for the traditional broadcast environments (where requests do not leave until they are served) [16]. The algorithm proposed here is essentially identical to that in [16], except for one crucial difference, as elaborated below.

In the proposed algorithm, whenever the server needs to determine the item to be transmitted next, it executes the 5 steps listed below. The algorithm calculates a "weight" for each item according to a given weight function F_i (defined below) and returns the item with the maximum weight value. For future reference, let Q denote the time at which next item is to be transmitted, and let R_i denote the time at which item i was most recently transmitted.

Step 1. For each item i, $1 \leq i \leq M$, determine the value of weight F_i.
Step 2. Determine maximum F_i over all items. Let F_{max} denote the maximum value.
Step 3. Choose item j such that $F_j = F_{max}$. If this equality holds for more than one item, choose any one of them arbitrarily.
Step 4. Broadcast item j.
Step 5. Set the latest broadcast time of item j (R_j) equal to the current time.

The choice of F_i function in the above algorithm significantly affects its properties. In our past work, we have proposed an algorithm named $S2P$ that minimizes the mean waiting time for the traditional broadcast model (where requests do not leave) [16], and also an algorithm that minimizes the variance of waiting time [13] with the traditional model.

In this paper, we use the F_i definition below with the objective of designing a broadcast schedule that maximizes the service ratio and minimizes the waiting time of impatient users. Note that the right-hand side of the expression below is obtained by substituting s_i by $(Q - R_i)$ in the expression in Theorem 1.

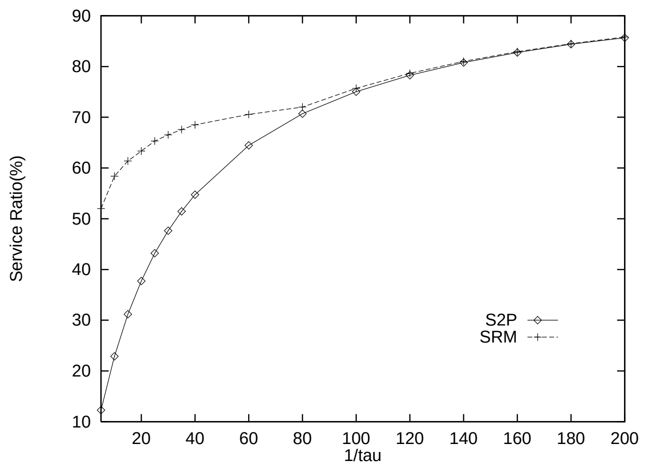
$$F_i = \frac{p_i}{l_i}(\tau(Q - R_i)e^{-\tau(Q-R_i)} + e^{-\tau(Q-R_i)} - 1)$$

Recall that Q is the time at which next item is to be transmitted, and R_i is the time when item i was most recently transmitted.

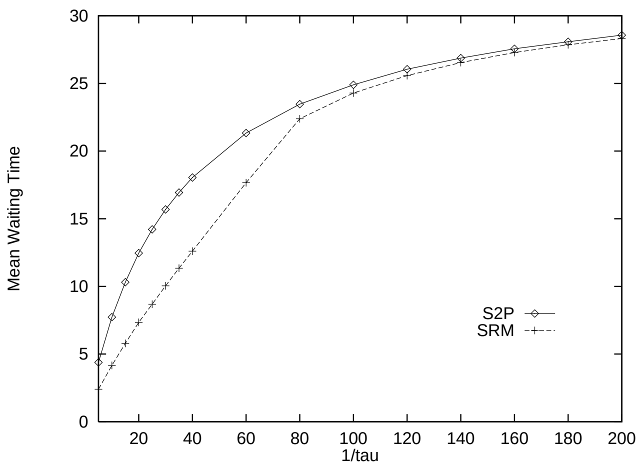
The proposed algorithm that uses the above definition of F_i will be referred to as the SRM (Service Ratio Maximized) algorithm. Note that the $S2P$ algorithm [16] that minimizes the mean waiting time in the traditional model uses $F_i = \frac{p_i}{l_i}(Q - R_i)^2$ as the weight function.

3 Simulation Results

Performance comparisons are made between the proposed SRM algorithm and the $S2P$ algorithm introduced in [16]. As noted before, the $S2P$ algorithm is designed to minimize the mean waiting time in an environment where requests do *not* leave until they are serviced. Of course, in the model under consideration in this paper, requests can leave before being serviced. The motivation behind comparing



(a) Service ratio



(b) Mean waiting time

Figure 1: Performance vs. Average User Patience Limit

SRM with $S2P$ is to determine how important it is (from a performance point-of-view) to consider the impact of user impatience.

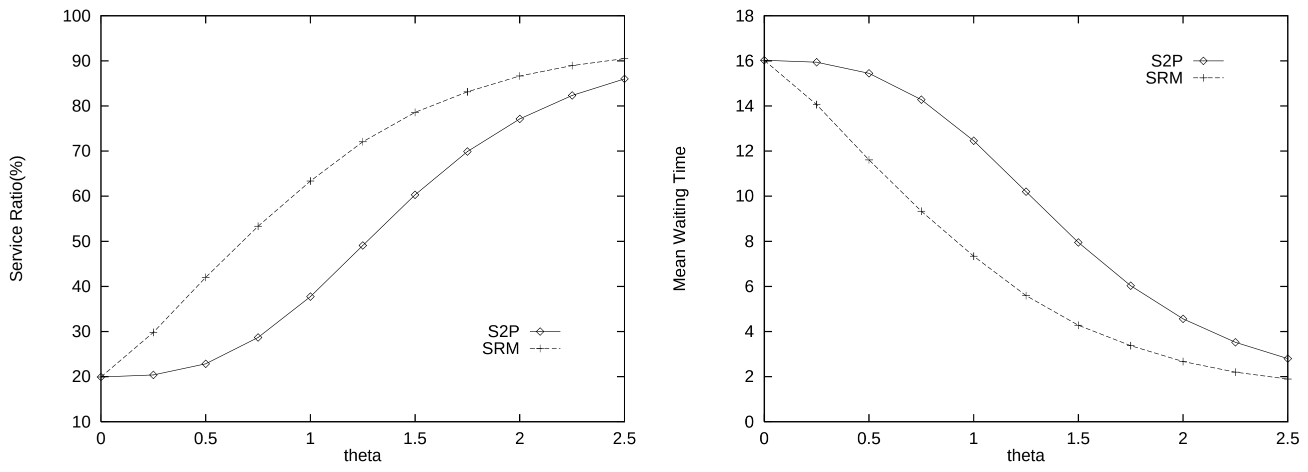
We consider a database consisting of 100 items, with $l_i = 1$ for all i (for lack of space, we omit results for other length distributions). The request probability p_i follows the Zipf distribution [18, 5, 15] with skew coefficient θ. Thus,

$$p_i = \frac{\left(\frac{1}{i}\right)^\theta}{\sum_{j=1}^{M}\left(\frac{1}{j}\right)^\theta}$$

For $\theta = 1$, Figure 1(a) plots the service ratio for the SRM and $S2P$ algorithms (as measured by simulation) versus $1/\tau$. Recall that $1/\tau$ is the average time a request will wait before leaving. It can be seen that, when users are tolerant of large waiting time (i.e., $1/\tau$ is large), the two algorithms yield similar service ratios. The SRM algorithm achieves significantly higher service ratio than $S2P$ for smaller $1/\tau$. As seen later, the service ratio is also a function of the skew coefficient θ.

For $\theta = 1$, Figure 1(b) plots the mean waiting time measured for the two algorithms, versus $1/\tau$. Again, the improvement achieved by SRM is higher for smaller values of $1/\tau$. Clearly, when $1/\tau$ becomes ∞ (i.e., requests do not leave unless serviced), the two algorithms should perform similarly, since the users are not impatient anymore. Actually, it can be shown that when $1/\tau = \infty$, the two algorithms produce identical broadcast schedules.

Next we examine the performance of the two algorithms as a function of skew in the data request pattern. The skew is characterized by parameter θ of the Zipf distribution – smaller θ results in smaller skew ($\theta = 0$ yields a uniform distribution), while larger θ results in larger skew.



(a) Service ratio (b) Mean waiting time

Figure 2: Performance vs. Data Request Skew

For $1/\tau = 20$ time units, Figures 2(a) and 2(b) plot service ratio and mean waiting time, respectively, versus skew coefficient θ for the SRM and $S2P$ algorithms. Observe that when θ (skew) is large, the two algorithms yield similar service ratio and mean waiting time. When user requests are targeted to a few data items, frequent transmission of the few "hot" items will serve most requests before they leave (and before they wait too long). This is the reason why both SRM and $S2P$ algorithms yield high service ratio and low mean waiting time with high θ.

When the skew coefficient θ is moderate (not too large nor too small), SRM algorithm yields better performance than $S2P$ algorithm – thus, with moderate θ, the impact of user impatience is significant.

When θ approaches 0, the two algorithms again perform comparably. When $\theta = 0$, both algorithms are reduced to the so-called "flat" algorithm which broadcasts all data items in a round-robin style, and thus, uniformly allocates the bandwidth of broadcast channel among the data items.

4 Conclusions and Future Work

In this paper, we consider impatient users who may withdraw their requests before they are served. In such an environment, the objective of a broadcast scheduling algorithm should be two-fold: maximize the fraction of requests that are served, and also minimize the waiting time for the requests. Based on our analytical results, we proposed a new scheduling algorithm SRM that can generate schedules achieving this objective, significantly better than an existing algorithm ($S2P$) that does not take user impatience into account. Based on our experience with similar algorithms [16, 13], we believe that the proposed SRM algorithm performs near-optimally, however, we cannot prove this as yet.

Significant work is needed on the problem of impatient users. Some issues to be considered are as follows: (a) determining measures of optimal performance with which a proposed algorithm may be compared (so as to determine how well the algorithm performs compared to optimal), (b) consideration of other models for user patience, including assigning a "cost" to unserved requests, and (c) performance of pull-based systems. The proposed approach can be extended to pull-based systems by replacing the use of p_i in the SRM algorithm, with the number of requests (say, r_i) pending for item i at a given time.

Acknowledgment

We would like to thank Professor Amotz Bar-Noy for helpful discussions which led us to consider users who may leave before their requests are satisfied. This research is supported in part by Texas Advanced Technology Grant 010115-248.

References

[1] S. Acharya, M. Franklin, and S. Zdonik, "Prefetching from a broadcast disk," in *12th International Conference on Data Engineering*, Feb. 1996.

[2] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast disks - data management for asymmetric communications environment," in *ACM SIGMOD Conference*, May 1995.

[3] S. Acharya, M. Franklin, and S. Zdonik, "Dissemination-based data delivery using broadcast disks," *IEEE Personal Communication*, pp. 50–60, Dec. 1995.

[4] D. Aksoy and M. Franklin, "Scheduling for large-scale on-demand data broadcasting," in *Proc. of INFOCOM'98*, Apr. 1998.

[5] M. H. Ammar and J. W. Wong, "The design of teletext broadcast cycles," *Performance Evaluations*, vol. 5, Nov. 1985.

[6] M. H. Ammar and J. W. Wong, "On the optimality of cyclic transmission in teletext systems," *IEEE Transactions On Communications*, vol. 35, pp. 68–73, Jan. 1987.

[7] S. Banerjee and V. O. K. Li, "Evaluating the distributed datacycle scheme for a high performance distributed system," *Journal of Computing and Information*, vol. 1, May 1994.

[8] L. J. Corwin and R. H. Szczarba, *Multivariable Calculus.* Marcel Dekker, Inc., 1982.

[9] J. Gescei, *The Architecture of Videotex Systems.* Prentice Hall, 1983.

[10] G. Herman, G. Gopal, K. C. Lee, and A. Weinrib, "The datacycle architecture for very high throughput," in *Proc. of ACM SIGMOD*, 1987.

[11] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Energy efficient indexing on air," in *International conference on Management of Data*, May 1994.

[12] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Data on the air - organization and access," *IEEE Transactions of Data and Knowledge Engineering*, July 1996.

[13] S. Jiang and N. H. Vaidya, "Response time in data broadcast system: Mean, variance, and trade-off," in *Workshop on Satellite Based Information Services (WOSBIS), Dallas, TX*, Oct. 1998.

[14] J. Shanmugasundaram, A. Nithrakashyap, J. Padhye, R. Sivasankaran, M. Xiong, and K. Ramamritham, *Transaction Processing in Broadcast Disk Environments.* Kluwer Academic Publishers, 1997.

[15] C.-J. Su and L. Tassiulas, "Broadcast scheduling for information distribution," in *Proc. of INFOCOM'97*, Apr. 1997.

[16] N. H. Vaidya and S. Hameed, "Data broadcast in asymmetric wireless environments," in *Workshop on Satellite Based Information Services (WOSBIS), Rye, NY*, Nov. 1996.

[17] Z. Zdonik, R. Alonso, M. Franklin, and S. Acharya, "Are 'disks in the air' just pie in the sky?," in *IEEE Workshop on Mobile comp. System*, Dec. 1994.

[18] G. K. Zipf, *Human Behaviour and the Principle of Least Effort.* Addison-Wesley, Reading, MA, 1949.

A Appendix: Maximizing the service ratio function R

Expected service ratio R is a multivariable function of $s_1, s_2, \ldots, s_M$. Since the fraction of bandwidth allocated to item i is l_i/s_i, and the M items together consume the available bandwidth, the values of the M variables $s_1, s_2, \ldots, s_M$ are subject to the following constraint:

$$\frac{l_1}{s_1} + \frac{l_2}{s_2} + \ldots + \frac{l_{M-1}}{s_{M-1}} + \frac{l_M}{s_M} = 1 \tag{4}$$

To solve the problem of finding maximum of R subject to the above constraint, we use the method of Lagrange's multipliers [8]. We consider the function R' below, where ϕ is an unknown constant.

$$R' \;=\; R - \phi(\frac{l_1}{s_1} + \frac{l_2}{s_2} + \ldots + \frac{l_M}{s_M} - 1) \;=\; \left[\sum_{i=1}^{M} \frac{p_i}{\tau s_i}(1 - e^{-\tau s_i})\right] \;-\; \phi(\frac{l_1}{s_1} + \frac{l_2}{s_2} + \ldots + \frac{l_M}{s_M} - 1)$$

Next, we set all derivatives of R' to 0. That is,

$$\frac{\partial R'}{\partial s_1} = \frac{p_1}{\tau s_1^2}(\tau s_1 e^{-\tau s_1} + e^{-\tau s_1} - 1) - \frac{\phi l_1}{s_1^2} = 0 \tag{5}$$

$$\frac{\partial R'}{\partial s_2} = \frac{p_2}{\tau s_2^2}(\tau s_2 e^{-\tau s_2} + e^{-\tau s_2} - 1) - \frac{\phi l_2}{s_2^2} = 0 \tag{6}$$

$$\ldots$$

$$\frac{\partial R'}{\partial s_M} = \frac{p_M}{\tau s_M^2}(\tau s_M e^{-\tau s_M} + e^{-\tau s_M} - 1) - \frac{\phi l_M}{s_M^2} = 0 \tag{7}$$

$$\frac{\partial R'}{\partial \phi} = \frac{l_1}{s_1} + \frac{l_2}{s_2} + \ldots + \frac{l_M}{s_M} - 1 = 0 \tag{8}$$

Solving these equations to find the extremum point is difficult, but we can exploit their symmetry to obtain an useful result. From Equation (5), we get

$$\tau\,\phi = \frac{p_1}{l_1}(\tau s_1 e^{-\tau s_1} + e^{-\tau s_1} - 1)$$

Similarly, we have

$$\tau\,\phi = \frac{p_2}{l_2}(\tau s_2 e^{-\tau s_2} + e^{-\tau s_2} - 1)$$

$$\ldots$$

$$\tau\,\phi = \frac{p_M}{l_M}(\tau s_M e^{-\tau s_M} + e^{-\tau s_M} - 1)$$

Denoting $\phi\tau$ as a constant K, it follows that

$$\frac{p_i}{l_i}(\tau s_i e^{-\tau s_i} + e^{-\tau s_i} - 1) = K, \quad i = 1, 2, \ldots, M \tag{9}$$

The equation above gives a necessary condition that must be satisfied by the schedule vector at the extremum point where R is maximized. We can show that this is also a sufficient condition for the optimal point. Further details are omitted here for lack of space.