# Scheduling Web Requests in Broadcast Environments

Jianliang Xu
Hong Kong Baptist Univ.
xujl@comp.hkbu.edu.hk

Wang-Chien Lee
Penn State Univ.
wlee@cse.psu.edu

Jiangchuan Liu
Chinese Univ. of Hong Kong
ljc@cse.cuhk.edu.hk

## ABSTRACT

On-demand broadcast has been supported in the Internet to enhance system scalability. Unfortunately, most of existing on-demand scheduling algorithms did not consider the time constraints associated with web requests. This paper proposes a novel scheduling algorithm, called Slack Inverse Number of requests (SIN), that takes into account the urgency and productivity of serving pending requests. Trace-driven experiments demonstrate that SIN significantly outperforms existing algorithms over a wide range of workloads.

## Categories and Subject Descriptors

C.2.0 [**Computer Systems Organization**]: Computer-Communication Networks—*general*

## General Terms

Algorithms, Performance

## Keywords

Scheduling algorithms, on-demand broadcast, Web, time constraints

## 1. INTRODUCTION

The ever growing popularity of the Internet and the resultant slow responses perceived by users have given rise to vast research efforts on improving the performance of web accesses. As the system scale and user base continue to grow, there is an increasing demand for information providers to be capable of concurrently delivering a large amount of information to a huge number of users, especially in popular events such as elections and Olympics games. As a result, innovative delivery technologies, including satellite communications (e.g., DIRECWAY [4]), cable networks, and wireless networks (e.g., 3G), have been developed and deployed to provide shared broadband Internet accesses.

Different from traditional networks, a distinguished feature of these new technologies is that they naturally support *broadcast*, which satisfies *all* outstanding requests for the same object by a single transmission, leading to efficient use of shared bandwidth. In general, there are two broadcast approaches: *push-based broadcast* computes the broadcast program based on historical statistics; *on-demand broadcast* schedules broadcast items based on outstanding requests. While push-based broadcast is useful for certain applications (e.g., small databases with stable access patterns), on-demand broadcast is more widely used for dynamic, large-scale data dissemination like that in the Internet.


*WWW2004*, May 17–22, 2004, New York, New York, USA.
ACM 1-58113-912-8/04/0005.

In many situations, web requests are associated with time constraints. These constraints can be imposed either by the users or the applications. Consider a driver who queries a traffic information server to select one of several alternative routes at some point ahead [2]. Clearly, it is necessary for the server to provide the driver with the traffic information (e.g., which route is less congested) *before* he reaches that point; otherwise, the information is of no value to the driver. As another example, HTTP requests are normally associated with timeout values to prevent users from unlimited waiting when the web servers are heavily loaded. This paper focuses on on-demand broadcast with time constraints, which we shall refer to as *time-critical on-demand broadcast*.

A key issue in the design of an on-demand broadcast system is the *scheduling algorithm* used to select and broadcast requested items from outstanding requests. While there has been significant work on the development of on-demand broadcast scheduling algorithms (e.g., [3]), none of them considered the time constraints associated with requests. On the other hand, although time-critical scheduling algorithms have been investigated for unicast-based real-time systems and push-based broadcast systems (e.g., [2]), they are not applicable or not effective to on-demand broadcast systems. In this paper, we propose a low-complexity scheduling algorithm, called Slack Inverse Number of requests (SIN), for time-critical on-demand broadcast.

## 2. SYSTEM MODEL

As shown in Figure 1, we consider a satellite-based broadcast architecture that captures all essential components of a typical on-demand broadcast system [1]. In this architecture, a large group of clients retrieve web pages maintained by a server. The clients send requests to the server through an uplink channel. Each request is characterized by a 3-tuple: $< id, t, d >$, where $id$ is the identifier of the requested item, $t$ is the time of request, and $d$ is a *relative deadline*. The *absolute deadline* of a request is given by $t + d$, beyond which the receipt of the requested item is of no value to the client. The client monitors a downlink broadcast channel for the requested item until the item is broadcast or the lifetime of the request expires.

The server broadcasts data items chosen from feasible requests based on a scheduling algorithm. The primary goal of a scheduling algorithm is to satisfy as many requests as possible. This is best measured by *request drop rate*, which is defined as the ratio of the number of requests missing their deadlines to the total number of requests. The selected items are sent to the network controller for broadcast and the associated requests are removed from the service queue. For simplicity, all data items are assumed to have equal size.

## 3. PROPOSED SIN ALGORITHM

This section describes the design philosophy and operations of SIN. We start by illustrating the factors affecting the performance

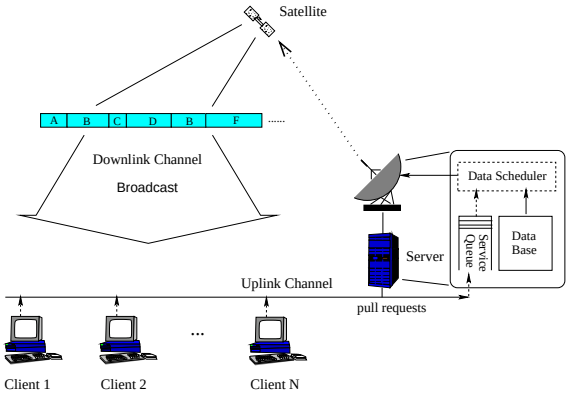


**Figure 1: On-Broadcast System Architecture**

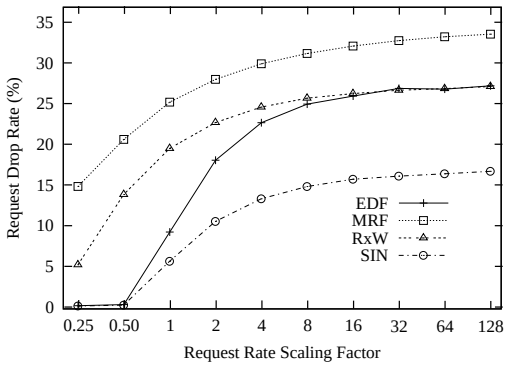


**Figure 2: Request Drop Rates for Request Arrival Rates**

of time-critical broadcast scheduling. The broadcast duration of an item is referred to as a *broadcast tick*. We compare EDF (earliest deadline first) and MRF (most requests first), two typical scheduling algorithms in unicast and broadcast respectively. At each broadcast tick, EDF broadcasts the item with the shortest remaining lifetime to cater for the *urgency* of requests. MRF, on the other hand, broadcasts the item that has the largest number of pending requests to account for the *productivity* of broadcasting. As shall be shown in Section 4, EDF and MRF respectively achieve good performance for certain workloads only. This motivates us to integrate the *urgency* and *productivity* factors to improve scheduling performance. Intuitively,

- For two items with the same number of pending requests, the one with closer deadline should be broadcast first.
- For two items with the same deadline, the one with more pending requests should be broadcast first.

Motivated by the above observations, we propose a new scheduling algorithm SIN. Specifically, the $sin$ value of each item that has at least one pending request is given by

$$sin = \frac{slack}{num} = \frac{1stDeadline - clock}{num},$$

where *slack* is the duration from the current time (i.e., $clock$) to the deadline of the most urgent pending request for the item (i.e., $1stDeadline$), and $num$ is the number of pending requests for the item. At each broadcast tick, the item with the minimum $sin$ value is broadcast on the downlink channel. A straightforward implementation of SIN is to compute, at each broadcast tick, the $sin$ values of all items that have pending requests. To reduce its complexity, we can maintain two lists, the S-list and the N-list, to index requested items. The S-list sorts the items in ascending order of $slack$, while the N-list indexes the items in descending order of $num$. The item with the minimum $sin$ value can be found efficiently by searching the two lists in an alternate fashion [1].

## 4. PRELIMINARY RESULT

A trace-driven simulator has been developed to evaluate the performance of the proposed algorithm. Real trace collected from the World Cup '98 website (i.e., the day-38 trace) was used to simulate the requests made by clients. The average request rate is 83 per second. To simulate different levels of workloads, we changed the time scale of the trace by introducing a *request rate scaling factor* $f$. The inter-arrival time between two consecutive requests was set to the actual time logged in the trace divided by $f$. The service rate (i.e., capacity) of the broadcast channel is described in the number of pages that can be transmitted per second. The default service rate was set at 10 pages/sec. To model the time constraints of requests, each request was assigned a relative deadline of $d$, which is different for different pages and uniformly distributed between 0 and 120 seconds.

We compare SIN with the existing algorithms EDF, MRF, and the recently proposed R$\times$W [1]. Figure 2 shows the request drop rates as a function of the scaling factor $f$. Comparing different scheduling algorithms, the proposed SIN algorithm performs the best throughout the tested range of scaling factor. The improvement of SIN relative to EDF and MRF is up to 38.0% and at least 49.8% respectively. Since MRF and R$\times$W ignore the request deadlines, their drop rates are high even when the system is lightly loaded (see the left part of Figure 2). In contrast, no requests are dropped by SIN and EDF at low system loads. When the system is heavily loaded (see the right part of Figure 2), SIN performs substantially better than both MRF and EDF.

## 5. CONCLUSIONS AND FUTURE WORK

On-demand broadcast has attracted a lot of attention in web content distribution due to its scalability to user population. An effective yet simple scheduling algorithm called SIN that integrates the urgency and productivity of items has been proposed in this paper. To the best of our knowledge, this is the first study to investigate the problem of scheduling time-critical on-demand broadcast. Trace-driven simulation experiments show that the proposed SIN algorithm considerably outperforms existing algorithms over a wide range of workloads. Some related results can be found in [5]. Ongoing research efforts include investigating the relative weight of the two factors in SIN, energy-efficient scheduling algorithms, and data staging issues.

## 6. REFERENCES

[1] D. Aksoy and M. Franklin. R$\times$W: A scheduling approach for large-scale on-demand data broadcast. *IEEE/ACM Trans. Networking (TON)*, 7(6):846–860, Dec. 1999.

[2] J. Fernandez and K. Ramamritham. Adaptive dissemination of data in time-critical asymmetric communication environments. In *Proc. Euromicro Real-Time Systems Symp.*, 1999.

[3] V. Liberatore. Multicast scheduling for list requests. In *Proc. IEEE INFOCOM'02*, June 2002.

[4] Hughes Network Systems. DIRECWAY homepage. Website at http://www.direcway.com/.

[5] J. Xu, X. Tang, and W.-C. Lee. Time-critical on-demand broadcast: Algorithms, analysis, and performance evaluation. Tech. Report, Hong Kong Baptist Univ., June 2003.