On Transmission Scheduling in a Server-less Video-on-Demand System[1]

C. Y. Chan and Jack Y. B. Lee

Department of Information Engineering
The Chinese University of Hong Kong
{cychan2, yblee}@ie.cuhk.edu.hk

Abstract. Recently, a server-less video-on-demand architecture has been proposed which can completely eliminate costly dedicated video servers and yet is highly scalable and reliable. Due to the potentially large number of user hosts streaming video data to a receiver for playback, the aggregate network traffic can become very bursty, leading to significant packet loss at the access routers (e.g. 95.7%). This study tackles this problem by investigating two new transmission schedulers to reduce the traffic burstiness. Detailed simulations based on Internet-like network topologies show that the proposed Staggered Scheduling algorithm can reduce packet loss to negligible levels if nodes can be clock synchronized. Otherwise, a Randomized Scheduling algorithm is proposed to achieve consistent performance that is independent of network delay variations, and does not require any form of node synchronization.

1 Introduction

Peer-to-peer and grid computing have shown great promises in building high-performance and yet low cost distributed computational systems. By distributing the workload to a large number of low-cost, off-the-shelve computing hosts such as PCs and workstations, one can eliminate the need for a costly centralized server and at the same time, improve the system's scalability. Most of the current works on grid computing are focused on computational problems [1], and on the design of the middleware [2]. In this work, we focus on another application of the grid architecture – video-on-demand (VoD) systems, and in particular, investigate the problem of transmission scheduling in such a distributed VoD system.

Existing VoD systems are commonly built around the client-server architecture, where one or more dedicated video servers are used for storage and streaming of video data to video clients for playback. Recently, Lee and Leung [3] proposed a new server-less VoD architecture that does not require dedicated video server at all. In this server-less architecture, video data are distributed to user hosts and these user hosts cooperatively server one another's streaming requests. Their early results have shown

[1] This work was supported in part by the Hong Kong Special Administrative Region Research Grant Council under a Direct Grant, Grant CUHK4211/03E, and the Area-of-Excellence in Information Technology.

that such a decentralized architecture can be scaled up to hundreds of users. Moreover, by introducing data and capacity redundancies into the system, one can achieve system level reliability comparable to or even exceeding those of high-end dedicated video servers [4].

Nevertheless, there are still significant challenges in deploying such server-less VoD systems across the current Internet. In particular, Lee and Leung's study [3] did not consider the issue of network traffic engineering. With potentially hundreds or even thousands of nodes streaming data to one another, the aggregate network traffic can become very bursty and this could lead to substantial congestion at the access network and the user nodes receiving the video data. Our study reveals that packet loss due to congestion can exceed 95% if we employ the common first-come-first-serve algorithm to schedule data transmissions.

In this study, we tackle this transmission scheduling problem by investigating two transmission scheduling algorithms, namely Staggered Scheduling and Randomized Scheduling. Our simulation results using Internet-like network topologies show that the Staggered Scheduling algorithm can significantly reduce packet loss due to congestion (e.g. from 95.7% down to 0.181%), provided that user nodes in the system are clock-synchronized using existing time-synchronization protocols such as the Network Time Protocol [5]. By contrast, the Randomized Scheduling algorithm does not need any form of synchronization between the user nodes albeit does not perform as well. Nevertheless, the performance of the Staggered Scheduling algorithm will approach that of the Randomized Scheduling algorithm when the network delay variation or clock jitter among nodes are increased. In this paper, we present these two scheduling algorithms, evaluate their performance using simulation, and investigate their sensitivity to various system and network parameters.

2 Background

In this section, we first give a brief overview of the server-less VoD architecture [3] and then define and formulate the transmission scheduling problem. Readers interested in the server-less architecture are referred to the previous studies [3-4] for more details.

2.1 Server-less VoD Architecture

A server-less VoD system comprises a pool of fully connected user hosts, or called nodes in this paper. Inside each node is a system software that can stream a portion of each video title to as well as playback video received from other nodes in the system. Unlike conventional video server, this system software serves a much lower aggregate bandwidth and thus can readily be implemented in today's set-top boxes (STBs) and PCs. For large systems, the nodes can be further divided into clusters where each cluster forms an autonomous system that is independent from other clusters.

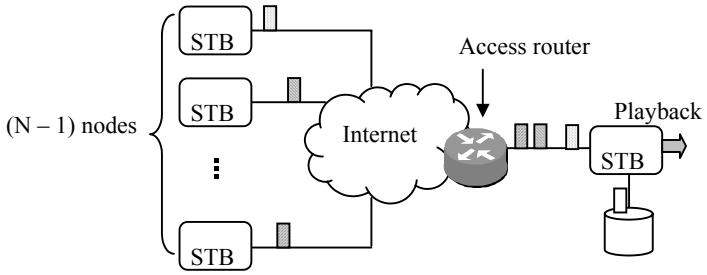

Fig. 1. A *N*-node server-less video-on-demand system

For data placement, a video title is first divided into fixed-size blocks and then equally distributed to all nodes in the cluster. This node-level striping scheme avoids data replication while at the same time share the storage and streaming requirement equally among all nodes in the cluster.

To initiate a video streaming session, a receiver node will first locate the set of sender nodes carrying blocks of the desired video title, the placement of the data blocks and other parameters (format, bitrate, etc.) through the directory service. These sender nodes will then be notified to start streaming the video blocks to the receiver node for playback.

Let N be the number of nodes in the cluster and assume all video titles are constant-bit-rate (CBR) encoded at the same bitrate R_v. A sender node in a cluster may have to retrieve video data for up to N video streams, of which $N - 1$ of them are transmitted while the remaining one played back locally. Note that as a video stream is served by N nodes concurrently, each node only needs to serve a bitrate of R_v/N for each video stream. With a round-based transmission scheduler, a sender node simply transmits one block of video data to each receiver node in each round. The ordering of transmissions for blocks destined to different nodes becomes the transmission scheduling problem.

2.2 Network Congestion

In an ideal system model, video data are transmitted in a continuous stream at a constant bit-rate to a receiver node. However, in practice data are always transmitted in discrete packets and thus the data stream is inherently bursty. In traditional client-server VoD system this problem is usually insignificant because only a single video server will be transmitting video data to a client machine and thus the data packets will be transmitted at constant time intervals. By contrast, video data are distributed across all nodes in a server-less VoD system and as a result, all nodes in the system participate in transmitting video data packets to a node for playback. If these data transmissions are not properly coordinated, a large number of packets could arrive at the receiver node's access network at the same time, leading to network congestion.

For example, Fig. 2 depicts a straightforward transmission scheduler - On Request Scheduling (ORS), which determines the transmission schedule based on the initial request arrival time. Specifically, a node transmits video data in fixed-duration rounds,

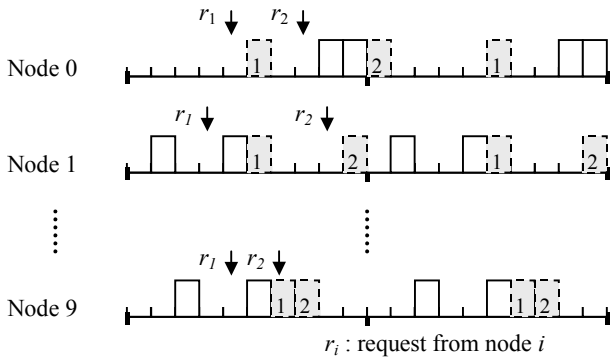

Fig. 2. Transmission schedules generated by the On Request Scheduling algorithm

with each round further sub-divided into N timeslots. The node can transmit one Q-byte data packet in each time slot. Let T_r be the length of round and $T_s = Q/R_v$ be the length of a timeslot, then with a video bit-rate R_v we can compute T_r and T_s from $T_r=NT_s=NQ/R_v$.

When a node initiates a new video streaming session, it will send a request to all nodes in the system. A node upon receiving this request will reserve an available timeslot in a first-come-first-serve manner to begin transmitting video data for this video session. For example, consider the scenario in Fig. 2 where there are 10 timeslots per round. Request r_1 reaches node 0, and is assigned to slot 5. On the other hand, when request r_2 reaches node 0 the upcoming slot has already been assigned to another stream and in this case the request will be assigned to the first available slot (i.e. slot 0). Note that for simplicity we do not consider disk scheduling in this study and simply assumed that data are already available for transmission.

It is easy to see that this ORS algorithm can minimize the startup delay experienced by end users as well as spread out data transmissions destined for different receivers to reduce burstiness of the aggregate network traffic leaving a node. While this algorithm may work well in traditional client-server VoD systems, its performance is unacceptably poor in a server-less VoD system. In our simulation of a 500-node system with Q=8KB and R_v=4Mbps, this ORS algorithm can result in over 95% packet losses due to congestion in the access network.

The fundamental problem here is due to the very large number of nodes in the system and the fact that data transmissions are packetized. With the ORS algorithm, a new video session will likely be assigned to timeslots that are temporally close together. Thus once transmission begins, all nodes in the system will transmit video data packets to the receiver node in a short time interval, and then all cease transmission for T_r seconds before transmitting the next round of packets. While the average aggregate transmission rate is still equal to the video bit-rate, the aggregate traffic is clearly very bursty and thus leads to buffer overflows and packet drops at the access network router connecting the receiver node.

Fig. 3. Transmission schedules generated by the Staggered Scheduling algorithm

3 Transmission Scheduling

To tackle the network congestion problem discussed earlier, we investigate in this section two transmission scheduling algorithms, namely the Staggered Scheduling (SS) and the Randomized Scheduling (RS) algorithms.

3.1 Staggered Scheduling (SS)

As mentioned in Section 2.2, the ORS algorithm can reduce the burstiness of the network traffic leaving a sender node, but the combined traffic from multiple sender nodes can still be very bursty. The fundamental problem is that the ORS algorithm attempt to schedule a video session to nearby timeslots in all nodes and thus rendering the combined traffic very bursty.

This observation motivates us to investigate a new scheduler – Staggered Scheduling, which schedules a video session to nodes in non-overlapping timeslots as shown in Fig. 3. Specifically, the timeslots are all pre-assigned to different receiver nodes. For node i serving node j data will always be transmitted in timeslot $(j–i–1) \bmod N$. For example, in Fig. 3 node 9 is served in timeslot 8 in node 0 while it is served in timeslot 7 in node 1. Thus the timeslot assignment of a video session forms a staggered schedule and hence the name for the algorithm.

Assuming the nodes are clock-synchronized, then transmissions from different nodes to the same receiver node will be separated by at least T_s seconds, thus eliminating the traffic burstiness problem in ORS. Nevertheless, the need for clock-synchronization has two implications. First, as clocks in different nodes cannot be precisely synchronized in practice, the performance of the algorithm will depend on the clock synchronization accuracy. Second, depending on the application, the assumption that all nodes in the system are clock-synchronized may not even be feasible. We investigate the former issue in Section 4.4 and tackle the latter issue in the next section.

Table 1. Initial assignment of the parameters of the data loss problem model

Parameters	Values
Cluster size	500
Video block size	8KB
Video bitrate	4Mbps
Access network bandwidth	$1.1R_v$
Router buffer size (per node)	32KB
Mean propagation delay	0.005s
Variance of propagation delay	10^{-6}
Mean router queueing delay	0.005s
Variance of clock jitter	10^{-6}

3.2 Randomized Scheduling (RS)

Staggered Scheduling attempts to perform precise control of the data transmissions to smooth out aggregate network traffic. Consequently, close synchronization of the nodes in the system is essential to the performance of algorithm. In this section, we investigate an alternative solution to the problem that does not require node synchronization.

Specifically, we note that the fundamental reason why aggregate traffic in ORS is bursty is because data transmission times of all the sender nodes are highly correlated. Thus if we can decorrelate the data transmission times then the burstiness of the traffic will also be reduced. This motivates us to investigate a new Randomized Scheduling algorithm that schedules data transmissions for a video session in random timeslots. Moreover, the randomly assigned timeslot is not fixed but randomized in each subsequent round to eliminate any incidental correlations.

It is easy to see that under Randomized Scheduling, one does not need to perform clock synchronization among nodes in the system. Each node simply generates its own random schedule on a round-by-round basis. We compare the performance of Staggered Scheduling and Random Scheduling in the next section.

4 Performance Evaluation

We evaluate and compare the three scheduling algorithms studied in this paper using simulation. The simulator simulates a network with 500 nodes. To generate a realistic network topology, we implement the extended BA (EBA) model proposed by Barabási et al. [6] as the topology generator, using parameters measured by Govindan et al. [7].

To model access routers in the network, we assume an access router to have separate buffers for each connected node. These buffers are used to queue up incoming data packets for transmission to the connected node in case of bursty traffic. When the buffer is full, then subsequent arriving packets for the node will be discarded and thus resulting in packet loss.

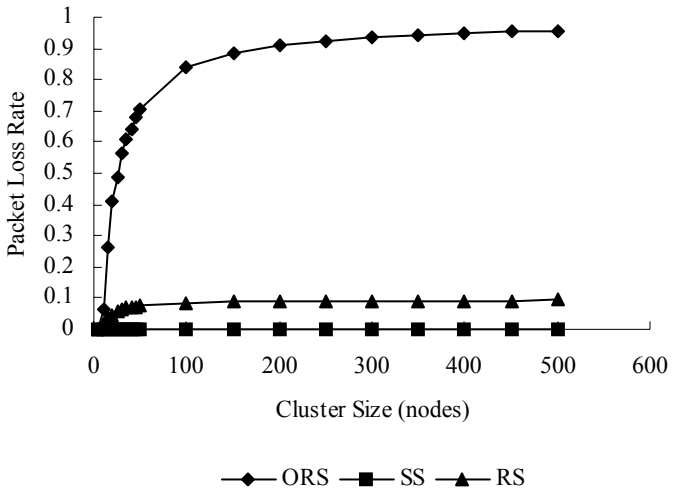

Fig. 4. Comparison of packet loss rate versus cluster size for ORS, SS, and RS

To model the network links, we separate the end-to-end delay into two parts, namely, propagation delay in the link and queueing delay at the router. While the propagation delay is primary determined by physical distance, queueing delay at a router depends on the utilization of the outgoing links. We model the propagation delay as a normally distributed random variable and the queueing delay as an exponentially-distributed random variable [8].

To model clock synchronization protocol, we assume that the clock jitter of a node, defined as the deviation from the mean time of all hosts, is normally-distributed with zero mean. We can then control the amount of clock jitter by choosing different variances for the distribution.

Table 1 summarizes the default values of various system parameters. We investigate in the following sections the effect of four system parameters, namely cluster size, router buffer size, clock jitter, and queueing delay on the performance of the three scheduling algorithms in terms of packet loss rate. Each set of results is obtained from the average results of 10 randomly generated network topologies.

4.1 Packet Loss Rate versus Cluster Size

Fig. 4 plots the packet loss rate versus cluster size ranging from 5 to 500 nodes. There are two observations. First, the loss rate decreases rapidly at smaller cluster size and becomes negligible for very small clusters. For example, for a 10-node cluster the loss rate is only 6.6%. This confirms that the traffic burstiness problem is unique to a server-less VoD system where the number of nodes is typically large. Second, comparing the three algorithms, On Request Scheduling (ORS), Staggered Scheduing (SS), and Randomized Scheduling (RS), ORS performs extremely poorly

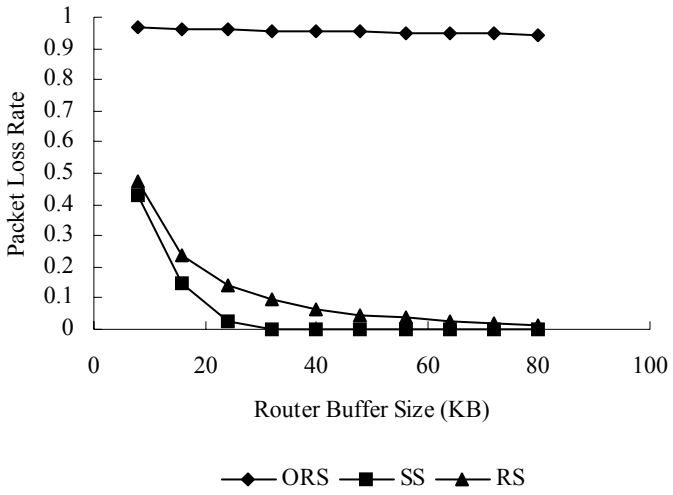

Fig. 5. Comparison of packet loss rate versus router buffer size for ORS, SS, and RS

with loss rates as high as 95%, which is clearly not acceptable in practice. RS performs significantly better, with a loss rate approaching 9.3% when the cluster size is increased to 500. Finally, the SS algorithm performs best with 0.18% packet loss regardless of the cluster size, demonstrating its effectiveness in eliminating bursts in the aggregate traffic.

4.2 Packet Loss Rate versus Router Buffer Size

Clearly, the packet loss rate depends on the buffer size at the access router. Fig. 5 plots the packet loss rate against router buffer sizes ranging from 8KB to 80KB. The packet size is Q=8KB so this corresponds to the capacity to store one to ten packets. As expected, the loss rates for all three algorithms decrease with increases in the router buffer size. In particular, the performance of RS can approach that of SS when the router buffer size is increased to 80KB. Nevertheless, ORS still performs very poorly even with 80KB buffer at the routers and thus one cannot completely solve the problem by simply increasing router buffer size.

4.3 Packet Loss Rate versus Queueing Delay

On the other hand, delay variations in the network can also affect performance of the schedulers. To study this effect, we vary the routers' mean queueing delay from 0.0005 to 5 seconds and plot the corresponding packet loss rate in Fig. 6.

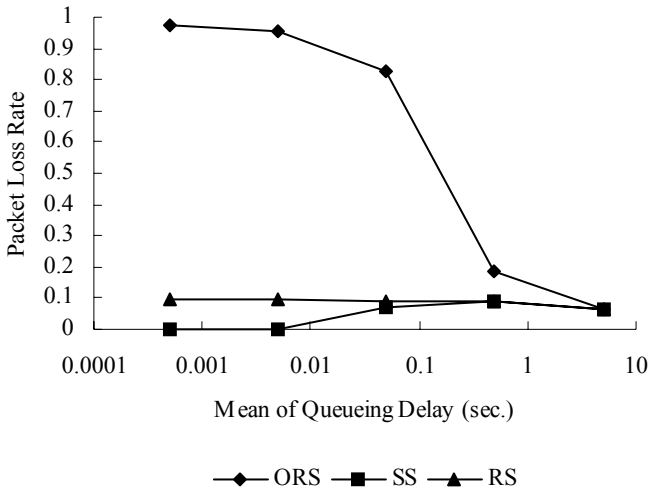

Fig. 6. Comparison of packet loss rate versus router queueing delay for ORS, SS, and RS

There are two interesting observations from this result. First, performance of the RS algorithm is not affected by changes in the mean queueing delay. This is because packet transmission times under RS are already randomized, and thus adding further random delay to the packet transmission times has no effect on the traffic burstiness.

Second, surprisingly performances of all three algorithms converge when the mean queueing delay is increased to 5 seconds. This is because when the mean queueing delay approaches the length of a service round (i.e. T_r=8.192 seconds), the random queueing delay then effectively randomize the arrival times of the packets at the access router and hence performances of both the ORS and SS algorithms converge to the performance of the RS algorithm.

This significance of this results is that transmission scheduling is effective only when random delay variations in the network are small compared to the service round length. Moreover, if the amount of delay variation is unknown, then the RS algorithm will achieve the most consistent performance, even without any synchronization among nodes in the system.

5 Conclusions

We investigated the transmission scheduling problem in a server-less VoD system in this study. In particular, for networks with comparatively small delay variations and with clock-synchronization, the presented Staggered Scheduling algorithm can effectively eliminates the problem of traffic burstiness and achieve near-zero packet loss rate. By contrast, the Randomized Scheduling algorithm can achieve consistent performance despite variations in network delay. More importantly, Randomized Scheduling does not require any form of node synchronization and thus is most

suitable for server-less VoD systems that do not have any centralized control and management. Since the problem is defined in the packet level, we would expect these results can be easily applied to both stream-based and object-based media, given that the requested media is packed and sent in the scheduled slots.

References

1. A. Oram, *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, O'Reilly Press, USA, 2001.
2. M. Baker, R. Buyya, and D. Laforenza, "The Grid: International Efforts in Global Computing," *International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet*, Rome, Italy, 31 July, 2000.
3. Jack Y. B. Lee and W. T. Leung, "Study of a Server-less Architecture for Video-on-Demand Applications," *Proc. IEEE International Conference on Multimedia and Expo.*, Lausanne, Switzerland, 26-29 Aug 2002.
4. Jack Y. B. Lee and W. T. Leung, "Design and Analysis of a Fault-Tolerant Mechanism for a Server-Less Video-On-Demand System," *Proc. 2002 International Conference on Parallel and Distributed Systems*, Taiwan, 17-20 Dec, 2002.
5. D. L. Mills, "Improved algorithms for synchronizing computing network clocks," *IEEE/ACM Transaction on Networks 3*, June 1995, pp.245-254.
6. R. Albert, and A.-L. Barabási, "Topology of Evolving Networks: Local Events and Universality," *Physical review letters*, vol.85, 2000, p.5234.
7. R. Govindan, and H. Tangmunarunkit, "Heuristics for Internet Map Discovery," *IEEE Infocom 2000*, Tel Aviv, Israel, Mar. 2000, pp.1371-1380.
8. D. Gross, and C. M. Harris, *Fundamentals of Queueing Theory*, 3rd ed. New York: Wiley.