A Super-Peer based P2P Live Media Streaming System

Poo Kuan Hoong, Hiroshi Matsuo *†

Abstract—**Peer-to-peer (P2P) file sharing has become increasingly popular, accounting for as much as 70% of Internet traffic by some estimates. Recently, we have been witnessing the emergence of a new class of popular P2P applications, namely, P2P audio and video streaming. While traditional P2P file distribution applications target elastic data transfers, P2P streaming focuses on the efficient delivery of audio and video content under tight timing requirements. In these applications, each node independently selects some other nodes as its neighbors and exchanges streaming data with neighbors. In this paper, we propose and investigate a full distributed, scalable, and cooperative protocol for live video streaming in an overlay peer-to-peer network. Our protocol, termed P2P Super-Peer based Unstructured Live Media Streaming (PALMS-SP), makes use of combination of push-pull scheduling methods to achieve high performance (in term of delay, stream continuity, cooperation, etc.). The main contribution of PALMS-SP is that it reduces the end-to-end streaming delay and in turn results better delivered quality. Furthermore, with the implementation of two-layer based overlay network that consists of super-peers and ordinary peers, PALMS-SP leverages on the heterogeneity of bandwidths and shows better Quality of Service (QoS). We have extensively evaluated the performance of PALMS-SP. Our experiments demonstrate that PALMS-SP achieves good streaming quality with the existence of super-peers.**

Keywords: peer-to-peer, streaming, overlay, push-pull, super-peer

1 Introduction

Peer-to-peer (P2P) file sharing has become increasingly popular, accounting for as much as 70% of Internet traffic by some estimates. Recently, we have been witnessing the emergence of a new class of popular P2P applications, namely, P2P audio and video streaming. While traditional P2P file distribution applications target elastic data transfers, P2P streaming focuses on the efficient delivery of audio and video content under tight timing requirements. Still in its infancy, both live and on-demand P2P streaming have the potential of changing the way we watch TV, providing ubiquitous access to a vast number of channels, personalizing your TV experience, and enabling roaming TV services. For a long time, traditional approaches that are client/server based e.g., Akamai [17] have been used for streaming multimedia applications over the Internet.

Over the past few years, P2P networks have emerged as a promising approach for distribution of multimedia content over a network. Some P2P network related research is by the following authors [8],[9],[12],[15],[16]. One form of P2P network, the peer-to-peer overlay, offer a promising approach to support one-to-many multimedia streaming applications without any special support from the network, called P2P streaming. The basic building blocks for P2P streaming, called *nodes* or *peers*, are no longer passive receivers of data but can act both as clients and servers at the same time. Stream data are simultaneously received, played, and passed to other connected peers. The goal of P2P streaming mechanisms is to maximize delivered quality to individual peers in a scalable fashion despite the heterogeneity and asymmetry of their access link bandwidth. An effective P2P streaming mechanism depends on the effective utilization of the outgoing bandwidth of most participating peers.

1.1 Motivation

In live P2P streaming, the media stream is a continuous flow of media data encoded from the streaming server. Media content generated must be delivered to participating nodes under a tight time constraint. Nodes should be able to receive media content before the playout deadline or the media content will be considered obsolete and discarded. The key challenges for a peer in P2P live media streaming applications include:

1. locate supplier peers with the desired media segments before the playout time deadline
2. choose peers that are likely to provide good performance for playback
3. manage parallel download and upload to connected neighbor nodes

*Nagoya Institute of Technology, Nagoya-shi, Showa-ku, Gokisocho, Aichi, JAPAN, 466-8555. Email:{khpoo@matlab., matsuo@}nitech.ac.jp

†This work was supported by the by the Ministry of Education, Culture, Sports, Science and Technology, Government of Japan, and also by a grant from the Hori Foundation Science Promotion Foundation, Japan.

4. re-transmission of lost packets before playout deadline
5. managing the connections with connected peers in the network due to the dynamicity of peers joining and leaving

In this paper, we propose and study the self-organizing, decentralized protocol capable of building and maintaining two-layer super-peer based, overlay topologies for P2P streaming live and non-interactive media streaming, called PALMS-SP (*P2P Super-Peer based Unstructured Live Media Streaming*). Similar to DONet [16], PALMS-SP is based on data-driven and receiver-based that is built on a super-peer based two-layer unstructured overlay media streaming. PALMS-SP is designed to operate in conditions where nodes have heterogenous bandwidths and resources. The existence of super-peers makes the networks to be more effective because they combine the efficiency of the centralized client-server model with the autonomy, load balancing, and robustness of distributed search. They also take advantage of the heterogeneity of capabilities across peers. Generally, *super-peers* are nodes that are faster and/or more reliable than ordinary nodes that take on server-like responsibilities and provide services to a set of *clients*. Super-peers allow decentralized networks to run more efficiently by exploiting heterogeneity and distributing load to machines that can handle the burden. On the other hand, this architecture does not inherit the flaws of the client/server model, as it allows multiple, separate points of failure, increasing the health of the P2P network. In comparison to DONet, which only employs pure pull method, PALMS-SP employs a combination of two methods for media streaming, namely the pull method and push method.

The key contributions of this paper are summarized as following:

- We propose a super-peer based two layer (super-peers and ordinary peers layers) P2P overlay network for live media streaming.
- We propose the combination push-pull based model instead of the commonly used pure pull based streaming mechanism in order to reduce the end-to-end delay.
- We formally define the push-pull data distribution scheduling problem.
- We propose a generic two-layer super-peer based system for scalable live media streaming system that incorporates swarm-like delivery with the combination of push-pull streaming to minimize latency observed by end users and maximize delivered quality.

The rest of the paper is organized as follows: Related work is discussed in Section II. In Section III, we discuss the overview system of PALMS-SP. In section IV, we present a generic system architecture for the implementation of PALMS-SP. We describe the details of the simulation setting and performance metrics in Section V. Section VI provides the result of performance evaluation based on simulation in various conditions. Finally we present our conclusions and discussion on future works in Section VII.

2 Related Work

The existing P2P streaming systems can be roughly classified into three main families:

Structured: In these systems, participating peers are organized into a hierarchical tree structure to form an application overlay network. Media content is pushed from the source towards all peers. Differences between systems of this family concern the way nodes are organized and algorithms used to build and repair the tree structure. The fundamental limitations of these systems are *(i)* the delivered quality to individual peers is limited by the minimum bandwidth among the upstream connections from the source *(ii)* a large fraction of outgoing bandwidth of peers that are leaves is not utilized. Some of the systems in this family are NICE [2], *End system Multicast* [4], PeerCast [6], and ZIGZAG [7].

Unstructured: The limitations of the structured family system have motivated a new approach where participating peers form a mesh-based overlay and incorporate swarm-like content delivery. This approach is inspired by file sharing protocols with swarm mechanisms such as BitTorrent [5]. Media content is broken by the source into chunks that are then available to participating peers. Nodes independently request and download the chunks they need to complete the stream. Systems like Chainsaw [13] and DONet [16] have presented a mesh-based P2P streaming mechanism that incorporates swarm-like media content delivery.

Other: These systems do not belong specifically to one of the previous families. Most of these systems have hybrid architectures that combine features of structured control overlay with unstructured media content delivery. Examples of such systems are Bullet [10] and SplitStream [3].

A distinct, but related problem regards roles that nodes may assume: original P2P systems were based in a complete "democracy" among nodes. The common assumptions of "everyone is a peer" is generally applied. However, physical hosts running P2P software are usually very heterogeneous in terms of computing, storage and communication resources, ranging from high-end servers

to low-end desktop machines. The super-peer paradigm is an answer to both issues. The super-peer approach to organize a P2P overlay is a trade-off solution that merges the client-server model relative simplicity and the P2P autonomy and resilience to crashes. The need for a super-peer network is mainly motivated by the fact to overcome the heterogeneity of peers deployed on the Internet. A super-peer connected with some ordinary peers has sufficient CPU power, bandwidth, and storage capacity and plays a role of a controller. A ordinary peer has the same ability of other ordinary peers have. Authors [14] proposed some design guidelines and fundamentals characteristics are discussed. A mechanism to split node clusters is proposed and evaluated analytically. Super-peer solutions proved to be effective solutions in the real world. Applications like Kazaa (FastTrack) [20] and Skype [23] are two outstanding examples. However, their actual protocols are not publicly available and thus it is difficult for other protocols to make comparison in terms of designs and performances.

The PALMS-SP protocol is a super-peer based two-payer P2P overlay network that focuses on the latency between peers and delivery ratio of live media streaming. We incorporate the work in [16] by considering a combination of push-pull methods, rather than pure pull methods for the streaming mechanism. Our main objective is to reduce the end-to-end delay and in turn enhances delivered video quality.

3 PALMS-SP : System Overview

PALMS-SP is based on data-driven and receiver-based unstructured two-layer super-peer based overlay media streaming. It is designed to operate in scenarios where the nodes have heterogenous and variable bandwidth resources. For the ease of exposition, we refer to the media source as the *streaming server* and receivers as *ordinary peer*. The term *peers* and *nodes* are interchangeable, and refer to the all the ordinary peers. We consider a network consisting a large collection of nodes. The network is highly dynamic; new nodes may join at any time, and existing nodes may leave, either voluntarily or by *crashing*.

PALMS-SP consists of three major components: (i) **overlay construction mechanism**, which organizes participating peers into an two-layer super-peer based overlay; (ii) **streaming scheduling mechanism**, which determines the delivery of content from the streaming source to individual nodes through the overlay; and (iii) **super-peer management mechanism**, which determines which nodes may switch role at will from a ordinary peer to super-peer status. In the following subsections, we describe these components in PALMS-SP.

3.1 Overlay Construction

In PALMS-SP, nodes are functionally identical. They are free to exchange control information and media content data from the stream. Each peer maintains a certain number of connected nodes that are known as *neighbors*. Each node can potentially communicate with every other node in the network. Each node receives media content from a certain number of neighbor nodes and relays the content to a certain number of neighbor nodes. Nodes are heterogenous: they differ in their computational, storage capabilities, and bandwidth. Nodes may act as super-peers or ordinary nodes. Each super-peer SP is associated with a *capacity* value $max(SP)$ that represents the maximum number of ordinary nodes associated to a super-peer SP.

The basic task of the overlay construction mechanism component for each node is to be in charge of finding appropriate super-peer and neighbors for each node through the gossip method so that the application layer network can be successfully built up. To join the streaming session, a new peer contacts the bootstrapping node, (streaming server in the case of PALMS-SP) to learn about super-peers and other participating peers upon arrival. Streaming server is selected as streaming server persists during the lifetime of streaming and its identifier/address is universally known. This could be regarded as the login process. The bootstrapping node returns a list of selected super-peers that can potentially serve as parent nodes. The new peer contacts these potential super-peers to determine whether they are able to accommodate a new child node. This is by determining whether the super-peer still has enough allocation slots on the outgoing degree. In the case of PALMS-SP, each peer is associated to exactly one super-peer. The number of child nodes associated to a super-peer is pre-determined. As shown in Figure 1, an overlay network consists of two layers, namely *ordinary peers layer* (lower) and *super-peer* (higher) layers. The ordinary peer and super-peer layers are composed of a set of ordinary peers and a set of super-peers, respectively. A collection of a super-peer SP and its ordinary peers $OP_1, OP_2, ...OP_n (n \geq 1)$, and it is referred to as a *cluster* C_{SP}. A super-peer SP_i is connected with another super-peer SP_j at the super-peer layer. The PALMS-SP topology can be summarized as the following:

- each node is either super-peer or a normal peer;
- each ordinary peer OP is associated to exactly one super-peer SP;
- the number of ordinary nodes associated to a super-peer SP does not exceed $max(SP)$.

In traditional super-peer networks shown in Figure 2, ordinary peers in a cluster cannot directly communicate

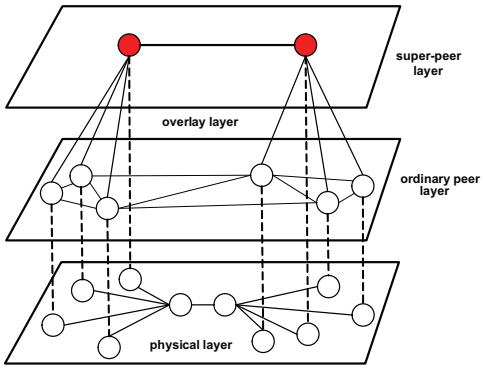


Figure 1: Two-layer overlay network composed of ordinary peer and super-peer layers

with each other. The ordinary peers have to communicate with each other through super-peer in the cluster. It takes at least two hops to delivery message from a ordinary peer to another ordinary peer. In this paper, we assume each ordinary peer can directly communicate with every neighbor peer in a cluster. Because of this assumption, the number of communication between a super-peer and its ordinary peers can be reduced and the super-peer has a lighter workload.

Each node has a member table that contains a list of neighbor nodes obtained from the super-peer. The information in member tables is encapsulated into a UDP packet and exchanged among neighbors periodically. Each node updates its member table in accordance with the member table sent by its neighbors. A super-peer SP holds all the information on service of every ordinary peer in a cluster C_{SP}. Each node sends a periodical *heartbeat* message to update its super-peer. If a node does not update its super-peer periodically, it will be removed from the member table. Once a node leaves, super-peer will broadcast a "leave message" to all its ordinary peers within its cluster. The nodes that receive this message will delete the respective node from its member table as well. Therefore, the failure of any neighbors can be detected by constantly monitoring periodical messages from super-peer.

In order to locate a better neighbor, which has higher uplink, a peer in PALMS-SP periodically replaces the neighbor with the least contribution by selecting nodes with higher scores (the ratio of uploaded packets over downloaded packets). This operation helps each node maintain a stable number of partners in the presence of node departures, and it also helps to discourage the existence of free riders within the network.

3.2 Streaming Scheduling

PALMS-SP employs a swarm-like content delivery mechanism that is similar to BitTorrent [5]. Nodes in the swarm protocol will be attracted to nodes that possess newly generated content. The main advantages of swarming content delivery are its ability to effectively utilize the outgoing bandwidth of participating peers and its robustness against the dynamics of peers arrival and departure, which is also known as *churn*.

The streaming scheduling mechanism of each node is responsible for exchanging packets with all its neighbors. Swarm-like content delivery is incorporated in PALMS-SP. Each peer periodically generates a report i.e., *buffer map* of its newly received packets and sends it to its neighbor nodes. Each peer periodically requests a subset of required packets from each neighbor node based on the reports received. The pull mode is deployed to fetch absent packets from its neighbor nodes and in turn tries its best to deliver packets requested by the neighbors. Packets requested by the pull mode are determined by the packet scheduling algorithm, which is much similar to the data-driven approach in DONet [16].

Every node also maintains a *window of interest*, which is the set of sequence packets that the node is interested in acquiring at the current time. Figure 3 illustrates the fundamental concept of the sliding window. A sliding *window of availability* contains the list of segments available for each node. This is the information for the *buffer map* shared with other neighbor nodes. The node slides its window of interest forward over time as new packets stream in. If a packet has not been received by the time it "falls off" the trailing edge of the window, the node will consider that packet lost or obsolete and will no longer try to acquire it. Figure 4 shows the buffer state of a node at a given time.

To accommodate the bandwidth heterogeneity among peers, the content is encoded with Multiple Description Coding (MDC). Basically, MDC organizes the streaming content into several sub-streams where each sub-stream can independently decoded. The use of MDC for video

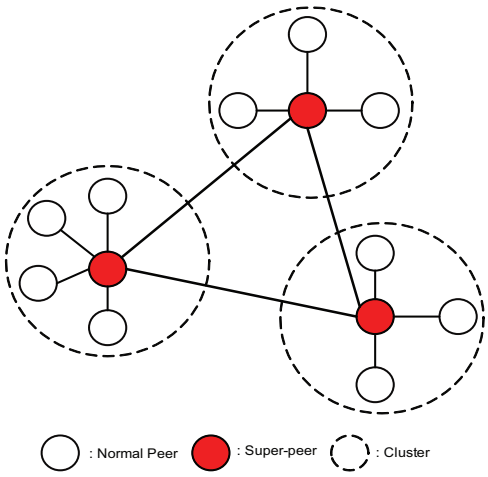


Figure 2: Traditional super-peer network

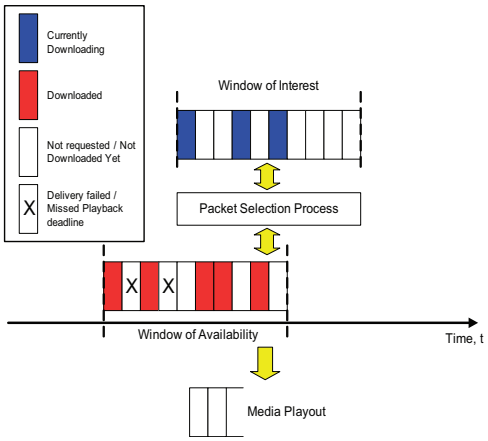


Figure 3: Data buffer for PALMS-SP node

streaming has been widely studied. Padmanabhan *et al.* propose that introducing redundancy can provide robustness in media streaming [12]. The delivered quality to each peer is proportional to the number of independent sub-streams it receives. With MDC coding, each peer is able to receive the proper number of sub-streams that are delivered through the combination push-pull streaming mechanism.

3.3 Super-Peer Management Mechanism

At the super-peer layer, a super-peer is connected with other super-peers in a flat unstructured overlay network. Super-peer selection problem is highly challenging because in the peer-to-peer environment, a large number of super-peers must be selected from a huge and dynamically changing network in which neither the node characteristics nor the network topology is known priori. Thus, simple strategies such as random selection don't work. Super-peer selection is more complex that classic *dominating set* and *p-centers* from graph theory, known as the NP-hard problems, because it must respond to the dynamicity of nodes join and leave (churn) and function in an environment that is highly heterogeneous. PALMS-SP employs a simple heuristic protocol for super-peer selec-

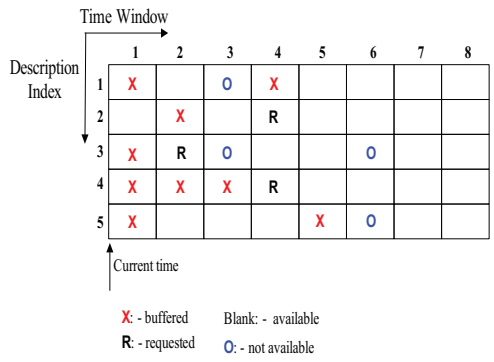


Figure 4: Buffer state of a node

Input:
$bw[k]$: bandwidth from neighbor k
$num\_suppliers$: number of suppliers
$bm[i]$: buffer map of connected node i
$deadline[j]$: deadline of packet j
$expected\_set$: set of packets to be pulled
$set\_neighbors$: number of neighbors of the node

Scheduling :
for packet $j \in expected\_set$ **do**
Make $num\_suppliers = 0$
for $l \in \{1..set\_neighbors\}$
• Calculate T_j, Time for Transmitting packet j :
$T_j = deadline[j]$ - $current\_time$
$num\_suppliers = num\_suppliers + bm[l, i]$
end for
end for
if $num\_supplier$=1
• Potential supplier = 1
for $j \in \{1..$expected_set$\}$
$supplier[j] \leftarrow \arg_r\{bm[r, i]=1\}$
end for j
else
• Potential Suppliers > 1
for $j \in \{1..$expected_set$\}$
for $r \in \{1..$num_suppliers$\}$
• Find r with the highest $bw[r]$ and enough available time $t[r, j]$
$supplier[j] \leftarrow \arg_r\{$ bw[r] > bw[r'],
t[r',j] > t[r,j],$r,r' \in set\_suppliers\}$
end for
end for
end if
Output $supplier[j]$
Do Pull packets from $supplier[j]$
Do Update *Buffer Map*

Figure 5: Pull Method Heuristic Algorithm

tion. We adopt the super-peer selection protocol which is similar to the H_2O (Hierarchical 2-level Overlay) [11] protocol for super-peer selection. H_2O is an advertisement-based super-peer selection protocol that is deployable in an unstructured overlay network.

The best know example of super-peer selection in a peer-to-peer application is the *gnutella* [19] protocol for selection of ultrapeers - peers with sufficient bandwidth and processing power to serve as proxies for other peers. The use of ultrapeers reduces network traffic and speeds up content delivery. In gnutella, any peer can select itself as an ultrapeer if it meets the following requirements : it has been up for at least 5 minutes, has high bandwidth, sufficient processing power, able to handle a large number of simultaneous TCP connections, and if not behind

any firewall or NAT. The ultrapeer selection protocol dynamically adjusts the number of super-peers as follows: if a leaf peer cannot find an ultrapeer with free slots, it can promote itself to be an ultrapeer. Ultrapeers also can downgrade themselves when they are no longer serve as any leaf nodes, or through negotiation with nearby peers. In term of cluster size, there is a tradeoff between aggregate and individual load. It is good to choose a cluster size that is small enough to keep a reasonable individual load and provide reliability to the system, but large enough to avoid the knee in aggregate load when cluster size is small.

The basic idea behind super-peers management mechanism for PALMS-SP is simple and intuitive. Ordinary peers with similar locality e.g., *IP addresses* are connected to the same super-peer. At the initial stage, all nodes start as ordinary peers. Nodes may switch role at will. The decision process is completely decentralized. An ordinary peer selects one super-peer to send queries and share resources. Since the ordinary peer depends on super-peer's capabilities, the ordinary peer should select the super-peer which can provide it with the best service. There are many metrics that may be used to select the best super-peer, such as average response time, bandwidth, processing capabilities, storage and so on. These metrics may have different weights depending on the objective. For PALMS-SP, we focus on response time, bandwidth and processing capabilities. In order to be selected as super-peer, ordinary peer must obtain reasonable scores for all the metrics. A super-peer can switch back to ordinary peer role only when a super-peer has lost all its clients due to nodes leaving or crashing. Super-peers exchange connected ordinary peers information at the super-layer layer. Information of connected ordinary peers is encapsulated into a UDP packet and exchanged among super-peers periodically.

4 Proposed System : PALMS-SP

The algorithms presented in this section make up the core of the PALMS-SP system. They determine how each node chooses its partner for data exchange, how data packets to be sent are chosen and scheduled, which data packets are to be requested from each connected neighbor and data to be pushed by connected super-peers.

4.1 Scheduling Algorithm

Given the buffer maps of a node and that of its partners, a schedule is to be generated for the pull and push mechanisms for fetching the expected packets from the partners and sending packets to connected neighbors. Basically, a simple heuristic algorithm is used for both pull and push mechanisms.

Input:
$set\_clients$: number of connected client nodes
$bm[i]$: buffer map of connected client node i
$deadline[k]$: deadline of packet k
$expected\_set$: set of packets to be pushed

Scheduling :
for packet $k \in expected\_set$ **do**
for $l \in \{1..set\_clients\}$
• Find Packet with the highest time-stamp :
$T_k = deadline[k]$ - $current\_time$
end for
end for
for $receiver \in \{1..set\_clients\}$
• Roulette Wheel Selection for receiver
end for
Output $receiver[k]$
Do Push packet to $receiver[k]$

Figure 6: Push Method Algorithm

4.1.1 Pull Mechanism

The main algorithms used for peer selection for pull and push mechanisms are an altruistic algorithm.

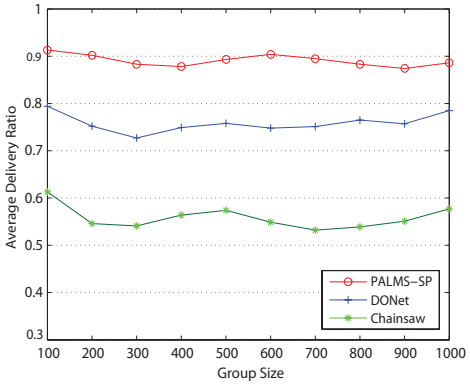
The algorithm for pull methods is similar to the heuristic used in DONet [16] and BitTorrent [5]. The main purpose of the pull method is to request the rarest packets among those that are locally available, and to distribute the request across different possible suppliers. The pull algorithm is shown in Figure 5.

Using the information gathered from the *buffer map* exchanged among neighbor sets, packets that are rarest across the neighborhood are requested with higher priority than more common ones. Packets with the same number of suppliers are randomly requested from one of the neighbors that can provide them. This is to limit the load on any single peer.

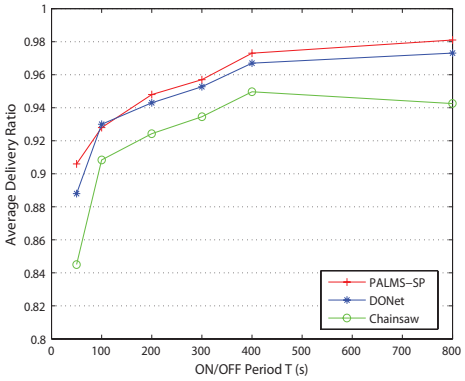
4.1.2 Push Mechanism

The push mechanism is the process of packet delivery by a super-peers to connected clients. Inspired by the work conducted by [1], the push mechanism for PALMS-SP employs two simple techniques too. Generally, the push mechanism consists of a proactive component where data packets are pushed forward by super-peer to connected clients, and a reactive mechanism where packets are pushed forward based NACKs information received.

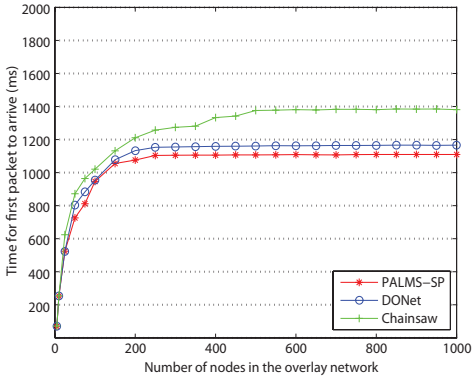
In order to increase delivery ratio, each super-peer at the super-peers layer, proactively send data packets to con-



(a) Delivery Ratio as function to Group Size



(b) Average Delivery Ratio as a function to ON/OFF Period, T(s). Group Size is 1000



(c) Average time for arrival first packet

Figure 7: Comparison Simulation Results

nected ordinary peers. The priority of data packets to be pushed is based on the *least frequently used (LFU) policy*. Moreover, due to the unreliability of the network link or a neighbor failure, some of the packets are lost during transmission. An overlay node can detect missing packet using gaps in the packet sequence numbers. This information is used to trigger NACK-based retransmission through the next interval of push mechanism for the super-peer. Thus, with the help of the push mechanism, packets are pushed and received at the receiver nodes at a second time interval. A good selection strategy is required to distribute the packets. This is to ensure that each super-peer pushes packets that are not close to the playout deadline and helps to reduce redundancy in push packets. Moreover, push packets should also take into account the NACK requests from connected nodes. The push algorithm is shown in Figure 6.

For the push packet scheduling, each super-peer tries to allocate packets that are least frequently used (LFU) into the *Push Packet Map*, PPm to be pushed. A *Push Packet Map*, PPm consists of node id and packet sequence number. A simple roulette wheel selection scheme is applied for the next time interval for each super-peer to push the available segments. Packets with the highest time-stamp or *least sent* will be given higher priority to be allocated into the *Push Packet Map*, PPm. Each super-peer keeps a counter of how many times each packet is sent. Packets with the least number of times sent will be chosen. In addition to that, packets that required retransmission based on NACKs received will be allocated into the *Push Packet Map*, PPm

5 Simulation Scenario

In this section, we perform extensive simulations to study the performance of PALMS-SP. Simulations on the algorithms' behavior test for under different user arrival/ departure patterns, different network sizes, bandwidth distributions, and streaming rates using network simulator ns-2 [21].

1) Video Data: The length of the video is 120 minutes (a typical length for a movie).

2) Video Coding: We used a video stream that is MDC encoded with 5 descriptions. For simplicity, we assume that all descriptions have the same constant bit rate of 100 Kbps. Therefore, the rate of the full quality version of the stream is 500 Kbps.

3) Peer Parameters: The incoming access link bandwidth for all peers are set to 500 Kbps. The incoming access links of all peers are set to 500 Kbps so that each peer can easily receive the full quality playback rate. The buffer length is set to 30 seconds. In all our experiments we use a *heartbeat* period of 5 seconds for all simulated protocols. The interval for the next round of push mechanism is set for every 5 seconds.

4) Network Topology: Topology is generated by using Georgia Tech Internetwork Topology Models (GT-ITM) generator [18]. The delay on the access links are randomly selected between 5 ms to 25 ms.

5) Performance Metrics: We use three basic Quality of Service (QoS) performance metrics, i.e., Average Delivery Ratio, Delivery Latency and Data Overheads.

6 Simulation Results

We have examined the impact of heterogenous bandwidths and different nodes arrival/departure patterns on the performance of PALMS-SP streaming. We also study the three metrics of interest: Delivery quality, Delivery latency and Data overheads. We compare the push-pull protocol performance of PALMS-SP with DONet [16] and

Chainsaw [13]. Both DONet and Chainsaw employ pure pull mechanism. DONet employs a rarest-first strategy as the block scheduling method, and select suppliers with the most surplus bandwidth and enough available time first. Chainsaw uses a purely random strategy to decide what blocks to request from neighbors.

Delivery Quality: Figure 7(a) shows the average delivery ratio for PALMS-SP in comparison to DONet and Chainsaw. We define ***delivery ratio*** to represent the number of packets that arrive at each node before playback deadline over the total of number of packets encoded. We set the streaming rate as 500kbps. From the result, we can observe that the performances for PALMS-SP and DONet remain almost the same when group size increases. This is an indication that the performance of swarming based protocols or data-driven protocols is not affected by group size. In other words, swarming protocols have a good scalability. However, Chainsaw method decreases more in comparison to PALMS-SP and DONet. As shown in Fig. 7(a), PALMS-SP has 20% gains compared to DONet and over 45% gains compared to Chainsaw.

We also tested the performances of PALMS-SP in comparison to DONet and Chainsaw under dynamic network environment. We set all the nodes to join in an initialization period of around 1 minute, and then we set each node changes its status according the ON/OFF model. The node actively participates the overlay during the ON period, and leaves (or fails) during the OFF period. Both ON and OFF periods are exponentially distributed. Figure 7(b) shows that a shorter ON/OFF period leads to a lower delivery ratio. However, the overall delivery ratio for PALMS is higher in comparison to DONet and Chainsaw because the additional push mechanism employed at the super-peer layer is able to help to recover from a vast majority of losses. Note that Chainsaw displays the poorest performances

Delivery Latency: In Figure 7(c) we show the distribution of latency experienced by data packets at the different overlay nodes. In this experiment, we measure the average time for first packet arrival for all simulated protocols. Note that all protocols suffer an increase in average time of first packet arrival, stabilize, then stay relatively constant with the number of nodes. The increase is well identified and is due to the implementation of swarming protocols for PALMS-SP and DONet. Moreover, with the existence of super-peers and push protocol, packets have higher chances to be delivered to connected clients at a shorter period of time.

Data Overheads: In this section, we compare the overheads of PALMS-SP to DONet. Figure 8 shows that PALMS-SP incurs very low additional data overheads in comparison to DONet. The control overheads at different overlay nodes increase log-arithmically with the increase

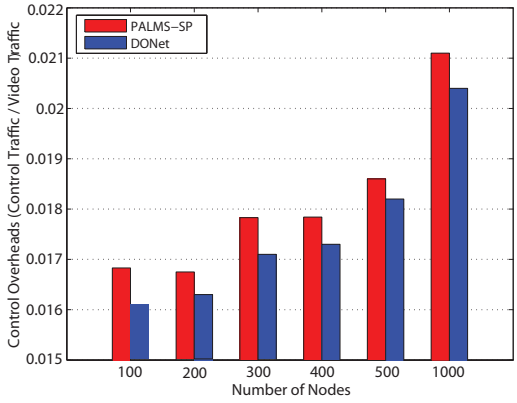


Figure 8: Comparison of Control Overheads for PALMS and DONet

in group size. The control overheads for PALMS-SP are slightly higher due to the additional messages such as *Push Packet Map* messages and NACKs. However the amount of increase at each overlay node is essentially minor, less than 3% of the total traffic. We believe the data overheads for PALMS-SP can be further reduced by increasing the window size.

7 Conclusions and Future Work

In this paper we presented PALMS-SP, a super-peer based P2P system for live media streaming. Our systems' innovative features are the usage of the combination push-pull protocol and the presence of super-peer two-layers that leverages on the heterogeneity of connected nodes.

To successfully deploy PALMS-SP streaming services, we proposed push-pull mechanism to address the issue of delivery quality and delivery latency. In this framework, the existence of super-peers improve delivered video quality by incorporating proactive and reactive push packets scheduling mechanism.

We evaluated the performance of PALMS-SP in comparison to DONet and Chainsaw. Our simulations conducted over ns2 demonstrated that PALMS-SP delivers quite a good playback quality even under formidable network conditions i.e., heterogeneity of network bandwidths, different user arrival/ departure patterns, different network sizes, and different streaming rates.

As part of our future plans, we aim to evaluate our proposed model, PALMS-SP in PlanetLab [22], in order to to further investigate the effectiveness and the robustness of our streaming model in a larger network and real network deployment. We are also keen in exploring various techniques to improve the delivery quality and delivery latency.

References

[1] S. Banerjee, S. Lee, B. Bhattacharjee and A. Srinivasan, "Resilient multicast using overlays," IEEE/ACM Transactions on Networking, pp. 237–248, vol. 14, no. 2, 2006.

[2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application layer multicast," SIGCOMM '02: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp. 205–217, August, 2002.

[3] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream : High-Bandwidth Multicast in Cooperative Environments," SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, pp. 298–313, Bolton Landing, NY. USA, October 2003.

[4] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," Measurement and Modeling of Computer Systems, pp. 1–12, June 2000.

[5] B. Cohen, "Incentives build robustness in BitTorrent," P2P Economics Workshop, Berkeley, CA, 2003.

[6] H. Deshpande, M. Bawa, and H. Garcia-Molina, "Streaming live media over peers," Tech. Rep. 2001-31, CS Dept, Stanford University, 2001.

[7] Duc A. Tran, Kien A. Hua, and Tai T, Do, "A Peer-to-Peer Architecture for Media Streaming." IEEE Journal on Selected Areas in Communications, vol. 22, no. 1, pp. 121–133, January, 2004.

[8] Y. Guo, K. Suh, and D. Towsley, "A Peer-to-Peer on-Demand Streaming Service and Its Performance Evaluation," ICME '03: Proceedings of the 2003 International Conference on Multimedia and Expo, pp. 649–652, Washington, DC, 2003.

[9] M. Hefeeda, A. Habib, R. Rotev, D. Xu, and B. Bhargava, "PROMISE: peer-to-peer media streaming using CollectCast," MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia, pp. 45–54, Berkeley, CA, November, 2003.

[10] D. Kostic, A Rodrigues, J. Albrecht, and A. Vahdat, "Bullet: high bandwidth data dissemination using an overlay mesh," SOSP '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, pp. 282–297, Bolton Landing, NY, October, 2003.

[11] V. Lo, D. Zhou, Y. Liu, C. GauthierDickey and Jun Li, "Scalable Supernode Selection in Peer-to-Peer Overlay Networks," HOT-P2P '05: Proceedings of the Second International Workshop on Hot Topics in Peer-to-Peer Systems, pp. 18–27, Washington, DC, USA, 2005

[12] V. N. Padmanabhan, H. J. Wang, and P. A. Chou, "Resilient Peer-to-Peer Streaming," ICNP '03: Proceedings of the 11th IEEE International Conference on Network Protocols, Atlanta, GA, pp. 16–27, November, 2003.

[13] Pai, V. Kumar, K. Tamilmani, K. Sambamurthy, V. Mohr, A. E., "Chainsaw: Eliminating Trees from Overlay Multicast," IPTPS, no. 3640, pp. 127-140, 2005.

[14] B. Yang and H. Garcia-Molina, "Designing a super-peer network," IEEE International Conference on Data Engineering (ICDE'03), pp. 49–60, Bangalore, India, 2003.

[15] S. Ye and F. Makedon, "Collaboration-aware peer-to-peer media streaming," MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia, pp. 412–415, NY, 2004.

[16] X. Zhang, J. Liu, B. Li, and T.P. Yum, "CoolStreaming/DONet : A data-driven overlay network for efficient live media streaming," INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, pp. 2102–2111, March, 2005.

[17] Akamai Technologies Inc, "Delivery a better Internet," [Online] Available: http://www.akamai.com/.

[18] E. W. Zegura, GT-ITM: Georgia Tech Internetwork topology models (software). http://www.cc.gatech.edu/project, 1996.

[19] Gnutella. (2003) The annotated Gnutella protocol specification v0.4. Online. Available: http://rfc-gnutella.sourceforge.net/developer/stable/index.html

[20] KaZaA.com. KaZaA. http://www.kazaa.com/us/index.htm.

[21] Network Simulator, Ns-2. http://www.isi.edu/nsnam/ns/

[22] The PlanetLab project, http://www.planet-lab.org/

[23] Skype - the Global Internet Telephony Company. http://www.skype.org/