Layered Peer to Peer Streaming Using Hidden Markov Models

Sheng-De Wang and Zheng-Yi Huang

National Taiwan University, Taipei, TAIWAN
{sdwang,d95921017}@ntu.edu.tw
http://www.ee.ntu.edu.tw/

Abstract. A fundamental problem in peer-to-peer streaming is how to select peers from a large network to request their media data. Due to the heterogeneity and the time-varying features of shared resources between peers, an adaptive method is required to select suitable peers. In this paper, we use Hidden Markov Models (HMMs) to model each peer to reflect the variation of resources. Among peers with different HMMs, the one which produces the maximum observation probability is selected as the serving peer. Through simulation results, we show that the proposed algorithm can achieve a good streaming quality and low communication overhead. In addition to these characteristics, the proposed model also comes with the fairness property.

Key words: Pee to Peer, Streaming, HMM

1 Introduction

Ubiquitous computing is a trend of current technology and the emergence of peer to peer networking makes it possible to achieve it, since P2P renders each peer as a server which can offer service to others. Besides that, the peer-to-peer paradigm offers an alternative possibility for streaming media over the network due to an important inherent characteristic: resources are shared among peers. Peers that simultaneously function as both clients and servers share their resources, such as computing power, bandwidth, storages and contents with others. This important characteristic avoids dedicated replication servers altogether and hence it does not have the bottleneck of client/server streaming architecture. A peer-to-peer media streaming system is operated in a kind of play-while-downloading mode [2]. An element of diversity is that the local storage of each peer is leveraged as a cache-and-relay mechanism in which requesting peers request media data and cache the most recently played media data during streaming [5]. The cached content can then be relayed to later peers that request the same content. But the most distinct feature of all is that from peer-to-peer streaming systems, users not only enjoy the availability of media contents but also the high quality of media streaming. The media streaming quality depends on many factors, ranging from the characteristics of the streaming sources, such as link capacity, availability, accessible bandwidth and overlapping of paths from multiple sources to receivers [8]. It is obvious that the peer selection policy plays an important role in peer-to-peer streaming.

In peer-to-peer multimedia streaming, the peer selection problem is how a peer selects a subset of peers from the network and request data from those selected peers, such that the desired media content could be received before their scheduled playback time in order to obtain a better streaming quality. The importance of the peer selection problem comes from that different outcome of selection would cause the different received time of data since peers have different capability to deliver data. Moreover, the different received time of data would result in a different streaming quality since the data received time is an important factor that affects the availability of data when they are needed. The difficulty of the selection problem follows from the heterogeneity and uncertainty of peers. Peers have varying bandwidths, diverse computation power and different buffered media contents. In addition, peers might come and leave in an unpredictable fashion and their resources vary with time. Therefore, a static and unadaptable approach to the peer selection problem will not work in the peer-to-peer streaming environment.

In this paper, we use a hidden Markov model (HMM, [1]) to model the bandwidth usage in the peer selection problem. That is, Peers are modeled as HMMs in which states represent status of bandwidth usage. A hidden Markov model is defined by a five-tuple, λ= (N, A, B, K, π,). N is the number of states in the model. A is state-transition probability distribution which is related to the distribution of offered bandwidth of neighbors. B denotes the observation symbol probability which depends on the state of a peer. The observation probability would be higher in a rich-bandwidth state than in a deficit-bandwidth state. K defines observation symbol, {0, 1}. The observation symbol 1 denotes that a peer is able to satisfy a request, 0 otherwise. π is the initial probability for each state. Before selecting a serving peer, a requesting peer would establish HMMs for some peers from its neighbors. Wish having a good streaming quality, the requesting peer initially set the observation O={1} with $|O|$ is arbitrarily in length and select a peer with maximum $P(O|\lambda)$, the highest observation probability, as the serving peer. Based on a layered streaming model, peers use the HMM streaming algorithm would obtain a high streaming quality and the fairness of the network. Moreover, the communication overhead incurred from the HMM streaming model is smaller as compared with other selection strategies.

The rest of this paper is organized as follows. Section 2 presents the assumed streaming model and our previous layered streaming model which the HMM streaming algorithm based on. Section 3 describes our solution to peers selection using hidden Markov model. Parameters of a HMM is also defined in detail in this section. Section 4 presents the simulation results. Related works are described at Section 5. Finally, Section 6 is the conclusion and our future work.

2 Layered Streaming

In our streaming model, we use layered coding to generate the media content. Briefly, layered coding mechanisms generate a base layer and n enhancement layers. The base layer is necessary for the media stream to be decoded, enhancement layers are applied to improve stream quality. The base layer can be decoded independently; however enhancement layers must be decoded based on subordinate layers. That is, the first en-

hancement layer depends on the base layer and each enhancement layer $j+1$ depends on its subordinate layer j, $1 \leq j \leq n-1$. To simplify the notation, we call the base layer as layer 1. When we mention an n layer media streaming architecture, it is known that 1 to n layers are all taken into account. And we use R_n to denote the playback rate of an n layer media content. A media content is partitioned into sequential segments which are coded with layered coding. A segment is the unit for requesting, sending and playback.

In addition to layered coding, peers are structured into layers according to the bandwidth they offer to the network. We use $L(P_i)$ to denote the layer of peer i, and is defined by

$$L(P_i) = j \text{ if } (R_j) \leq Bandwidth_o < (R_{j+1})$$

For peer P_i of layer j, the offered bandwidth (or called up-streaming bandwidth) of P_i must at least be larger than R_j. For peer P_i of layer j, the largest layer of media it can request is restricted to j. Each peer is assigned a priority based on layer number. The larger the layer number, the higher the priority it has. When two or more peers contend with the bandwidth of a sending peer, the higher priority peer would have the right to use the resource first.

3 Peer Selection Using HMM

In this section, we describe how to use hidden Markov model to solve the peer selection problem. First, we define a set of peers for each requesting peer P_r that can serve as serving peers. This set is related to layers of neighbors of P_r and its own layer number.

Definition 1. *Let $S(P_r)$ be the set of selectable peers of P_r such that P_r could request data from those peers and we have the following definition*

$$S(P_r) = \{P_s : L(P_r) \leq L(P_s)\} \text{ for all } P_s \in P_r\text{'s neighbors}$$

When P_r performs the peer selection, P_r could only select peers from $S(P_r)$ as serving nodes.

Since each peer P_s is of limited bandwidth, and bandwidth of P_s varies with time. Each requester, moreover, selects its senders in a distributed fashion. Therefore, it is possible that actual bandwidth of a selectable peer might not coincide on the local information known by a requesting peer. When such situation happens, the requester might not obtain its desired media content if the requested target can't afford enough bandwidth to satisfy requests and the streaming quality of requester, therefore, would decline. To solve this mismatch problem, variation of bandwidth should be taken into account. In this paper, we use HMM to solve this problem. Based on layered streaming model, HMM is used as a statistical model and use the result of HMM as a reference to select the most suitable sending peer. In the following content, P_r is known as a requesting peer who will request data from some other peers; P_s is known as some peer from the selectable set of P_r that P_r might select it as its serving node.

In our layered HMM streaming model, each P_s was modeled as a hidden Markov model which is defined by a five-tuple, $\lambda = (N, A, B, K, \pi)$. N is the number of states in the model. In our current work, two states are defined for each P_s that are named as *GOOD* state and *BAD* state, they will be abbreviated to G and B respectively. From

the perspective of P_r, P_s might be in either *GOOD* state or *BAD* state. If P_r assumes that P_s as in *GOOD* state, then P_r thinks P_s might have enough bandwidth to satisfy its request. Otherwise, P_r thinks P_s might be short of bandwidth. Since bandwidth of P_r varies with time, the state of P_r would also change with time and such variations can be defined by state-transition probability distribution, A = $\{a_{ij}\}$ where

$$a_{ij} = \ P(State_{t+1} = j | State_t = i) \text{ , i, j} \in \{GOOD, BAD\}$$

is the probability that state i at time t would transit to state j at time $(t+1)$. For example, a_{GB} represents the transition-probability from *GOOD* state to *BAD* state. Since in the our layered model, peers could only be requested by peers from lower layer and a priority mechanism is applied, the state-transition probability distribution A is, therefore, related to layer distribution of neighbors of P_s and the layer of P_r. Then A for P_s from the perspective of P_s is defined as

$$a_{BB} = a_{GB} = \ \frac{|C_{P_j}|}{|Neighbors_{P_s}|} \text{ , } C_{P_j} = \{P_j : L(P_r) < L(P_j) < L(P_s)\}$$

Once a_{BB} and a_{GB} are defined, a_{GG} and a_{BG} could be defined as

$$a_{GG} = a_{BG} = 1 - a_{BB}$$

$Neighbors_{P_s}$ denotes neighbors of P_s, C_{P_j} is a set that counts potential peers that might compete P_s with P_r. Hence, a_{BB} is the relative frequency that represents ratio of peers that might cause P_s to fail in receiving data. The more peers that might compete P_s with P_r, the more likely that P_s would stay in *BAD* state. Otherwise, fewer peer would compete with P_r and thus *GOOD* state would more precisely describe the actual bandwidth of P_s.

The observation symbols K, is defined as $K = \{0, 1\}$. The symbol 1 denotes that a request could be satisfied and 0 denotes otherwise. Hence, in order to get a best streaming quality, each requesting peer is expected to obtains an observation O = {1} such that |O| is as maximum as possible.

Since P_s might send data to P_r based on whether it can afford P_r's desired bandwidth or not, the observation probabilities are, therefore, dependent on states. The observation probability distribution B =$\{b_j(k)\}$ denotes the probability of each state to deliver data. $b_G(1)$ is the probability to satisfy a request when P_s staying at *GOOD* state. Similarly, $b_B(1)$ is the probability to deliver data when P_s staying at *BAD* state. Since *GOOD* state denotes that P_r would have higher probability to get desired data, therefore, the initial value of $b_G(1)$ would be between the range as

$$0.5 \leq b_G(1) \leq 1$$

Similarly, for *BAD* state of P_s, the initial value of $b_B(1)$ would be defined as

$$0 \leq b_B(1) \leq 0.5$$

Once $b_G(1)$ and $b_B(1)$ are defined, $b_G(0)$ and $b_B(0)$ could be defined as

$$b_G(0) = 1 - b_G(1) \text{ , } b_B(0) = 1 - b_B(1)$$

When P_r establishes HMM for P_s for the first time, P_r random choose a value between [0.5, 1] for $b_G(1)$ and a random value from the range [0, 0.5] for $b_B(1)$. The initial probability of each state is set to equal, namely $\pi_G = \pi_B = 0.5$. However, when P_r re-establishes HMM for P_s later, P_r would set $b_G(1)$, $b_B(1)$ and initial probabilities based on its history. If P_s could not satisfy P_r during the latest request, then P_r should set π_B higher than π_G and decrease the value of $b_G(1)$.

Algorithm 1 is the process to select the most suitable serving peer using HMM. A requesting peer first establishes HMM for each selectable peer and then chooses the one with the largest $P(O|\lambda)$ as the serving node. When P_r requests the streaming service, P_r will execute the algorithm for the first time. Then during streaming, when the streaming quality of P_r declines to a predefined threshold, P_r will re-execute the selection algorithm and adapt parameters of each HMM.

procedure SELECT
  $N \leftarrow 2$
  $K \leftarrow \{0, 1\}$
  $O \leftarrow \{1\}$ with $|O|$ random size
  for all $P_s \in S(P_r)$ **do**
    $a_{BB} = a_{GB} = \frac{|C_{P_j}|}{|Neighbors_{P_s}|}$
    $b_G(1) = Random(0.5, 1)$
    $b_B(1) = Random(0, 0.5)$
    $\pi_B = 0.5$
    if P_s can't deliver data to P_r latest request **then**
      increase a_{BB} and a_{GB}
      decrease $b_G(1)$
      increase π_B
    end if
    $a_{GG} = a_{BG} = 1 - a_{BB}$
    $b_G(0) = 1 - b_G(1)$
    $b_B(0) = 1 - b_B(1)$
    $\pi_G = 1\text{-}\pi_B$
    $A \leftarrow \{a_{BB}, a_{GB}, a_{GG}, a_{BG}\}$
    $B \leftarrow \{b_G(1), b_G(0), b_B(1), b_B(0)\}$
    $\pi \leftarrow \{\pi_G, \pi_B\}$
    $\lambda_{P_s} \leftarrow$ HMM(N, A, B, K, π);
  end for
  select λ_{P_s} with maximum $P(O|\lambda_{P_s})$
  return P_s
end procedure

Algorithm 1: Peer Selection Using HMM

4 EVALUATION

To investigate the performance of the proposed streaming model, we have carried out extensive simulations under various scenarios and the detailed experimental results are presented in this section. For each experiment, we report the mean value of results obtained through 10 runs with different network size: 10^4, $5x10^4$, 10^5 and $2x10^5$ respectively. Peers belong to one of following layers 1, 2, 3, 4, 5 and the distribution is of uniform distribution. To quantify the performance of media streaming system, we define the streaming quality of a peer as

$$Q = \frac{\sum_{i=0}^{\kappa-1} Z_i}{\kappa} \tag{1}$$

where κ is the number of segments of a media content and Z_i is a variable that takes value 1 if segment i arrives at the receiver before its scheduled playback time, and 0 otherwise. Thus value of Q would range from 0 to 1, where 1 is the best streaming quality and 0 is the poorest one.

To compare the performance with our HMM streaming model, additional two methods of peer selection are presented here. The first one is called OPT algorithm. When performing peer selection using the OPT algorithm, P_r probes the status of bandwidth of each selectable peer before selecting peer. Then P_r selects senders that are probed and their bandwidth would be at least larger than P_r's desired bandwidth, based on the collected information. It is obvious that such strategy would result in the best streaming quality since each requesting peer would have the up to date information of each selectable peer. However, OPT algorithm might suffered from heavily communication overhead. The second method to be compared with called Random algorithm, in which P_r selects senders from its selectable peer set randomly.

In the first simulation, streaming qualities of these three algorithms are compared. Figure 1 shows the compared results. OPT algorithm gets the best streaming quality. The quality of our algorithms is about 80% on average. In addition to its superior streaming quality, it can be shown that our algorithm is also scalable when the network size increases. Because Random algorithm selects peers in a random fashion without based on any information, it has the poorest streaming quality In the second simulation, we examine overhead of these three algorithms. Figure 2 shows the compared

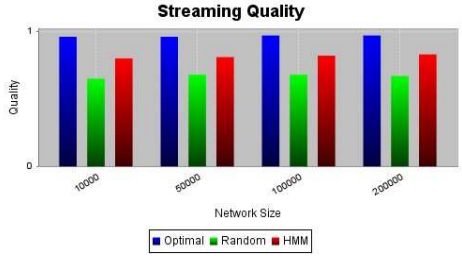

Fig. 1. Compared streaming quality with different selection methods

results of communication overhead. By overhead, it means that how many requests (or probing message)a requesting peer would issue during streaming. It also indicates that how accurate a selection algorithm would select a suitable peer. The lower the value, the more accurate the algorithm is, since such algorithm would have had selected the most suitable peer for requesting peers. As shown in Figure 2, the HMM streaming algorithm has the lowest overhead. And its overhead is also scalable as the network size is increasing.

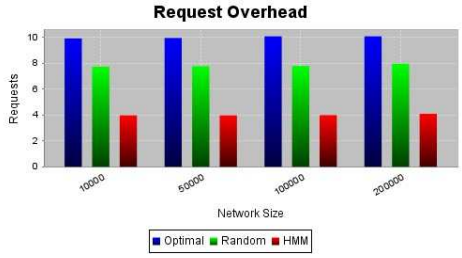

Fig. 2. Compared selection overhead with different selection methods

In the third simulation, we investigate differences of quality among different layers using the HMM streaming algorithm. Figure 3 shows the compared result among five layers. As shown from the result, the highest layer, the fifth layer, has the highest streaming quality and the lowest layer, the first layer, has the lowest quality on average. Such expectable result comes from that a priority mechanism is applied in the HMM streaming algorithm. It also shows that the proposed algorithm is endowed with fairness, since the more bandwidth offered to the network, the higher streaming quality would be obtained. The fourth simulation examines the impact of neighbors size using HMM

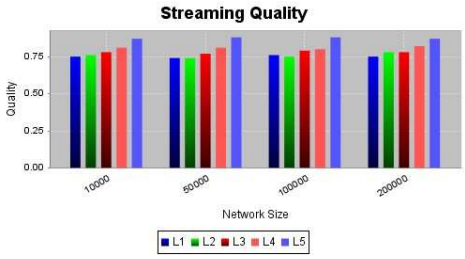

Fig. 3. Compared streaming quality with different layers using HMM streaming

streaming algorithm. Figure 4 shows the compared result of different neighbors size. Two different neighbors size are compared, namely 3% of the network size and 10% of the network size and the network size is of 10^5 peers. The size of neighbors affect how many selectable peers P_r would examine when performing algorithm, i.e. how many

HMMs would be calculate. As shown from the result, larger size of neighbors would result in a better streaming quality since there are larger HMMs would be examined and would provide a more precise selection. However, larger HMMs would contribute considerable computations when calculating observation probability.

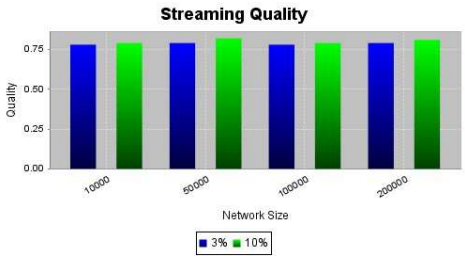

Fig. 4. Compact of neighbors size using HMM streaming

In the fifth simulation, we show the robustness of our approach when peer failures occur frequently. Figure 5 presents the compared result of different ratios of peer failure. When there is no failure in the network, the average streaming quality is the highest among all others. However, under the situation that peer failures are possible, the streaming qualities of different failure ratios are still stable figures. Such results come from that when a requesting peer detect that a sender can't supply data anymore, the requesting peer would decrease $b_G(1)$ and increase a_{BB}, a_{GB} and π_B of this sender, that would made this sender less chance to be selected when the requesting peer perform peer selection next time. Therefore, our approach is robust under the situation where peers may fail frequently.

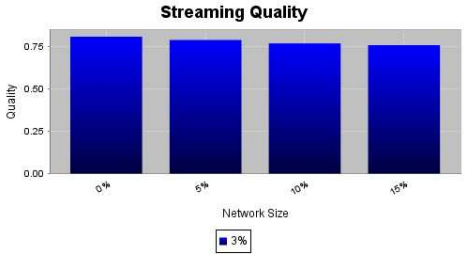

Fig. 5. Robustness of the HMM streaming model

5 Related Work

There are two possible mechanisms when mentioning layered media streaming, namely cumulative streaming and noncumulative streaming [15]. In this paper, we adopt the

cumulative layering approach in which the media data is encoded into one base layer and one or more enhancement layers. The base layer can be decoded independently, but enhancement layers are decoded cumulatively such that layer k can only be decoded when the layer 1 to the layer $k-1$ are decoded. In a non-cumulative streaming case, each layer is independent and the peer needs only subscribe to one layer. Peers in [16] decide which layers to request according to conditions on congestion and on spare capacity. PALS [17] allows requesting peers to orchestrate coordinated delivery by monitoring the overall throughput and periodically determining what is the target overall quality that can be delivered from all sending peers. K. Nahrstedt et al. [14] introduce requesting times of peers to determining a set of qualified sending peers for a requesting peer.

Other possible solutions to address the problem of peer-to-peer media streaming are by organizing network into structured hierarchical, one of them is called application level multicast(ALM). In an ALM-based streaming system, a multicast tree is constructed for media delivering over the network. Such multicast tree solves the peers selection in the sense that the requesting-sending relations are defined by the child-parent relations. However, how to build and maintain a multicast tree efficiently and with scalable control overhead is a critical issue. Nice [12] and Zigzag [11] both adopt hierarchical distribution trees in which peers are organized in a hierarchy of bounded size clusters but are fundamentally different due to their own multicast tree construction and maintenance strategies. SpreadIt [7] builds a single distribution tree in which a joining process is done by traversing the tree nodes downward from the source until reaching a node that is unsaturated and could accommodate the request. A deleting process is performed with a redirect process while a child detects the parent failure.

Three possible solutions to the peers selection are based on offered bandwidth of peers and take into account of network condition. B. Bhargava. et al. [2] proposes an optimal media data assignment algorithm which leads to the minimum buffering delay for a requesting peer. In their works, peers are classified into N classes according to the N possible values of their offered bandwidth to the network and data assignment is done under considering the available set of sending peer and the buffered size. B. Bhargava et al. [8] studies three possible peers selection techniques, namely random, end-to-end, and topology-aware with different goodness estimations for sending peers.

6 Conclusion

In this paper, we study the problem of peer selection and solve the problem using HMM based a layered streaming model. Simulation results show that our algorithm can obtain a superior streaming quality. Our algorithm is also endowed with fairness and scalability, these being important characteristics of peer to peer streaming. The fairness comes from the feature that the more bandwidth a peer contributes to the network, the higher streaming quality this peer will obtain. And the scalability comes from the fact that the streaming quality also remains at a superior level as the network size is increasing. The contributions of our work are:

– We present an HMM streaming model to solve the peer selection problem and define parameters of HMM to model the status of bandwidth.

– Our method is distributed and is with scalability and fairness, which are all important factor to a distributed network.
– High streaming quality would be obtained when using the HMM streaming model and only small communication cost is introduced by our method.

In our approach, however, each peer is simply classified into two states, namely *GOOD* and *BAD*, which is prone for P_r to misjudge a peer's state. In our future work, we will model states of peers as multi-states in order to get an exact prediction of bandwidth usage. And we will also find a statistic model to define the observation symbol probability distribution rather than rely on a random behavior. In addition to defining the initial values of parameters of HMM more precisely, we will also develop an algorithm to re-estimate these parameters when the issued request is failed to be satisfied in our future work.

References

1. Lawrence R. Rabiner. *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*. Proceedings of the IEEE, 77 (2), p. 257V286, February 1989.
2. D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava. *On peer-to-peer media streaming*. In Proc. of IEEE ICDCS'02, Vienna, Austria, July 2002.
3. Peer-to-Peer Membership Management for Gossip-Based Protocols Ayalvadi J. Ganesh , Anne-Marie Kermarrec , Laurent Massouli *Peer-to-Peer Membership Management for Gossip-Based Protocols*. IEEE Transactions on Computers, v.52 n.2, p.139-149, February 2003.
4. http://www.skype.com/
5. Changxi Zheng, Guobin Shen, Shipeng Li. *Distributed Prefetching Scheme for Random Seek Support in Peer to Peer Streaming Applications*. In Proc. ACM P2PMMS'05, November 2005.
6. Chuan Wu, Baochun Li. *Peer-to-peer tree construction: Optimal peer selection for minimum-delay peer-to-peer streaming with rateless codes*. In Proc. ACM P2PMMS'05, November 2005.
7. Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, Atul Singh. *SplitStream: high-bandwidth multicast in cooperative environments*. Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003, Bolton Landing, NY, USA.
8. M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava. *PROMISE: Peer-to-Peer Media Streaming Using CollectCast*. In Proc. of ACM Multimedia 2003, November 2003.
9. Habib, A.; Chuang, J. *Incentive Mechanism for Peer-to-Peer Media streaming*. Quality of Service, 2004. IWQOS 2004. Twelfth IEEE International Workshop on 7-9 June 2004 Page(s):171 - 180.
10. Meng Zhang, Li Zhao, Yun Tang, Jian-Guang Luo, Shi-Qiang Yang. *Large-Scale Live Media Streaming over Peer-to-Peer Networks through Global Internet* In Proc. of ACM P2PMMS'05, November 2005.
11. D. A. Tran, K. A. Hua, and T. Do. *ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming*. In Proc. of IEEE INFOCOM 2003, March 2003.
12. S. Banerjee, B. Bhattacharjee, and C. Kommareddy. *Scalable Application Layer Multicast*. In Proc. of ACM SIGCOMM 2002, August 2002.
13. H. Deshpande, M. Bawa, and H. Garcia-Molina. *Streaming Live Media over a Peer-to-Peer Network*, Technical report, Standford Database Group 2001-20, August 2001.

14. Y. Cui and K. Nahrstedt, *Layered peer-to-peer streaming*. In Proc. of ACM NOSSDAV, 2003.
15. T. Kim and M. Ammar. *A comparison of layering and stream replication video multicast scheme*. In International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), 2001.
16. S. McCanne, V. Jacobson and M. Vetterli. *Receiver-driven layered multicast*. In ACM SIGCOMM, 1996.
17. R. Rejaie and A. Ortega. *PALS: Peer to peer adaptive layered streaming* in NOSSDAV, June 2003.