International Journal of Multimedia and Ubiquitous Engineering
Vol.8, No.6, (2013), pp.377-386
http:// dx.doi.org/10.14257/ijmue.2013.8.6.37

An Efficient Supplier Side Scheduling Algorithm for P2P Live Streaming System

Hongyun Yang[1], Xiaoliang Zhu[1] and Xuhui Chen[2]

[1]National Engineering Research Center for E-learning, Central China Normal University, Hubei, 430079, China
[2] College of Mathematics & Computer Science, Wuhan Textile University, Hubei, 430070, China

yhycxh@gmail.com, zhuxl@mail.ccnu.edu.cn, wh.chxh@gmail.com

Abstract

Chunk scheduling is one of the key components in P2P streaming systems. Most of previous research works focus on receiver side's chunk/peer selection strategies and neglect the service order and available uplink bandwidth allocation problem at supplier side, which will cause the user's video quality descending under overloaded operating environments. In this paper, we propose the supplier side chunk priority model, formulate the supplier side scheduling problem as a linear programming problem and derive a greedy bandwidth resource allocation algorithm to solve it. The simulations demonstrate the proposed scheme effective comparing to the FCFS (First Come First Service) scheme.

Keywords: *Supplier side scheduler, P2P lives streaming, Bandwidth resource allocation, Relative urgency of playback*

1. Introduction

Nowadays the majority commercial Peer-to-Peer (P2P) live streaming systems adopt mesh-pull design to deliver video contents [1]. When designing data scheduling strategy, currently researchers mainly focus on receiver side chunk scheduler, which means that the receiver decides which chunk will be selected from which neighbor and neglect the design of the supplier side scheduling strategies. However, as we have known, there are two sides to make up an integrated scheduling process [2]. Neglecting the strategies of supplier side scheduling will cause the service response time increment in overloaded network environments.

In this paper, considering different requested chunks have different urgency of playback and even for the same chunk request, the urgency of playback is different due to the playback lags, we introduce the chunk priority model of supplier side, formulate the supplier side scheduling as linear programming problem, transform it as bandwidth allocation problem and propose a greedy algorithm to solve it. Our main contributions are as follows: first, we incorporate the relative urgency of playback and rarity of chunks into designing the priority model to achieve the tradeoff; second, we propose a greedy bandwidth allocation algorithm. The extensive experiments demonstrate the proposed scheme effective in improving the quality of experience of end users in overloaded operating environments.

The rest of this paper is organized as follows. Section 2 discusses some related works in this area. Section 3 formulas the supplier side scheduling problem as linear programming problem and proposes greedy algorithm to solve it in Section 4. Section 5 discusses our simulation methods and presents the evaluation results. Finally, Section 6 concludes our works.

2. Related Works

Recently, FCFS (First Come First Service) is the most widely used method adopted by the majority of current P2P streaming systems [3-5] for its simplicity and easy to realize in a direct or indirect way. However, this FCFS method neglects the urgency of playback and the rarity of requested chunks and will cause the supplier side can't response a large number of chunks requests in time, which further increases the wait time of peers and degrades the quality of experience of users.

Therefore, some studies take the rarity and urgency of playback of request chunks into account. In PULSE [6], supplier sides use "least sent first, Random" strategy to increase the data sharing scope. However it doesn't take the playback urgency of missing chunks into account and may cause the number of missing chunks increment. In LayerP2P[1], the supplier side peers maintain a different request queue for each receiver with regular and probing request types and apply the tit-for-tat-like strategy to determine the service selection probability. And the more related to our work, bin [7] proposes a priority-based supplier side scheduling scheme for a VOD (video On-Demand) system. In [7], the buffer is divided into urgent region and non-urgent region by pre-fetch window and greedy strategy and rarest first strategy adopted respectively when define the chunk model of supplier side scheduling. In this paper, considering that rarity and relative urgency of requested chunks playback for different receivers are two most important characters which affects the data scheduling methods at receiver side, we incorporate these factors to design the supplier side scheduling algorithm. So as to the receiver side scheduling strategy, we use latest useful chunk, random peer mechanism, which has been proved to achieve dissemination at an optimal rate and within an optimal delay [8].

3. Problem Formulation

3.1. Supplier Side Chunk Priority model

In a mesh-pull based P2P live streaming system, each media chunk has a playback deadline, which can be different from one peer to another by a few seconds or minutes. In addition, due to the deployment of buffering mechanisms, it is possibility of playback time lags among peers. So for the same missing chunk request from different peers, the urgency of playback is different (we call it relative urgency of playback).

In this paper, in order to improve the utilization of available uplink bandwidth of supplier peers, we model the chunk priority at supplier side as following. Firstly we introduce some definitions and notations, which are summarized in Table 1.

Table 1. Notations

Notation	Description	Notation	Description
U_p	The upload capacity of peer p, kbits/s	C_{ij}	The time of peer i sending the request of chunk j
<p,i>	The overlay link between peer p and peer i	m_i	The number of neighbors of peer i
U_{pi}	The maximum bandwidth capacity of link <p,i>	S_{ij}	The chunk of Number j in the buffer of peer i
u_{pi}	The available uplink bandwidth peer p allocates to peer i	g_{ij}^k	$g_{ij}^k=1$ represents chunk S_{ij} in the buffer of peer k, otherwise $g_{ij}^k=0$
NEI_i	peer i's set of neighbors	a_{ij}^k	$a_{ij}^k=1$ represents chunk S_{ij} in the sliding window of peer k, otherwise $a_{ij}^k=0$
α	playback urgency factor	h_{ij}	$h_{ij}=1$ represents peer i has received chunk j, otherwise $h_{ij}=0$
l_{ij}	The round trip time between peer i and peer j	v_{ij}^k	binary variable,
T	The service time interval	C	Chunk size, Kbits
W_T	The sliding window size, in seconds	m	The maximum length of request queue
d_{ij}	The playback deadline of chunk j on peer i	C_{ij}	The time of peer i sending the request of chunk j

According to the real-time requirement in live streaming, the priority of requested chunk is defined as formula (3.1).

$$P_{kj}{}^{q} = \begin{cases} 0, d_{qj} - C_{qj} - l_{qk} < 0 \\ \alpha \times \dfrac{W_T}{d_{qj} - C_{qj} - l_{qk} + 1} + (1-\alpha) \times \left(\dfrac{\sum_{i=1}^{m_k} (1 - g_{kj}^i) a_{kj}^i}{\sum\limits_{i \in NBI_k} h_{ij} + 1}\right), d_{qj} - C_{qj} - l_{qk} \ge 0 \end{cases} \qquad (3.1)$$

where P_{kj}^{q} represents the priority of requested chunk j sending from receiver q to supplier k. $d_{qj} - C_{qj}$ denotes the residual time before playback of the requested chunk j on peer q. $d_{qj} - C_{qj} - l_{qk}$ represents the maximum wait time of receiver q for chunk j, which is the serving deadline of the chunk request and equals the surplus time subtracting the round-trip delay between peer q and peer k. $\sum_{i=1}^{m_k} (1 - g_{kj}^i) a_{kj}^i$ denotes the times chunk j has been requested from neighbors during the period. $\sum\limits_{i \in NBI_k} h_{ij}$ represents the neighbor number of peer k which have received the chunk j. As shown in (3.1), P_{kj}^{q} is the sum of two terms. The first term represents the relative urgency of playback and the second term represents the value of the scheduled chunks, which uses the chunk request times and local scarcity ratio to estimate. And the urgency factor α satisfied $0 \le \alpha \le 1$.

The priority model of request chunks has the following characteristics: 1) for the receiver peer, the closer to the local current playback position, and the higher of the chunk's priority; 2) For the supplier peer, the chunks which are possessed by few neighbors and requested by more neighbors have higher priority. The model takes into account the different peers' play lag or semi-synchronized phenomenon, and the real-time and rarity characters. So in a certain extent, this scheme reduces the probability to send useless chunks to requested peers after the chunks' playback deadline and achieve high usage of available upload bandwidth of suppliers.

3.2. Supplier Side Scheduling Problem Formulation

We assume that a set of k peers N = {N_1,N_2,…, N_k}choose an existing peer p as the supplying peer. We use r_i={r_{i0},r_{i1},…,r_{im}}, ∀ 1 ≤ i ≤ k to represent the requested chunks sequences sending from receiver peer i. v^k_{ij} is a binary random variable that depends on whether the requested chunk is served. v^k_{ij} is defined as follows :

$$v_{ij}^{k}=\begin{cases}1,\text{if supplier peer i sends chunk j to request peer k}\\ 0,\text{ otherwise}\end{cases} \quad (3.2)$$

For each supplier peer, the goal is to maximize the sum of priority value of all served chunks, while not violating the constraint on peer and link bandwidth. Given the above definitions, the supplier side scheduling problem can be formulated as follows[7]:

$$z_p=\max\sum_{i=1}^{k}\sum_{j=1}^{m}p_{pj}^{i}v_{pj}^{i} \quad (3.3)$$

$$\text{s.t.} \sum_{i=1}^{k}u_{pi}\le U_p \quad (3.3a)$$

$$u_{pi}\le U_{pi},\forall 1\le i\le k \quad (3.3b)$$

$$\sum_{j=1}^{m}v_{pj}^{i}\le\frac{u_{pi}\times T}{C},\forall 1\le i\le k \quad (3.3c)$$

Equation (3.3) is the objective function of supplier peer p, which is to maximize the total priority value of the supplier peer p during the next service period. Equation (3.3a), (3.3b) and (3.3c) are the outbound bandwidth constraints. Equation (3.3a) denotes that the sum of allocated uplink bandwidth to receivers can't excess the outbound bandwidth of supplier peer p. Equation (3.3b) denotes the allocated uplink bandwidth from supplier peer p to receivers can't excess the maximum available link bandwidth between supplier peer and each receiver. Equation (3.3c) denotes the number of chunks served is limited by the link available bandwidth between the supplier and each receiver.

In fact, the problem the supplier side deals with chunks requests can be transformed as a bandwidth resource allocation problem. In the paper, we assume chunks have the same size. And the maximum number of chunks supplier peer p serves for receiver i is calculated as formula (3.4).

$$SC_{P\to i}=\sum_{j=1}^{m}v_{pj}^{i}\le\frac{u_{pi}\times T}{C}\le\frac{U_{Pi}\times T}{C} \quad (3.4)$$

The maximum number of service chunks during T service interval of supplier peer p can be calculated as follows: $SC_p = (U_p\times T)/C$. The problem of supplier side scheduling equals to the picking balls problem discussing as follows. Given there are k bags. Bag j has NM_j balls. The maximum number of balls picked out of bag j is M_j. The weight of each ball is already well-known. Let's W_{i1},W_{i2},….,W_{im}($1 \le m \le NM_i$) represent the weight of each ball in bag i. Then the problem can be transformed as how to pick SC_p balls so as to achieve the maximum weight. We use greedy strategy to achieve the optimal solution of the problem. First, we sort all the balls in k bags in descending order according to their weight, denotes as p^{sort}. Then we select the first SC_p balls from p^{sort} and class them according to their original bag number. Let Q_j denote the number of selected balls from bag j. Suppose Q_j is larger than M_j , we will discard the latter M_j-Q_j balls from bag j and select the following M_j- Q_j balls from p^{sort}. The process continues until SC_p balls are selected without violating the maximum constraints of each bag.

4. A Greedy Bandwidth Resource Allocation algorithm (GBRA)

The algorithm runs as follows. Through exchanging buffer map in period, peers know which chunks have been in its neighbors' buffer or not. When a peer receives several chunks requests from its neighbors, the peer launches the supplier side scheduler. First the supplier computes whether it is necessary to service the chunk request according to requested chunk's playback deadline, the sending time and transmission delay between the receiver and the supplier. Then after the coarse filtering, all the chunk requests are ranked according to their priority value computed as formula (3.1). And for every chunk request, it is pushed into respective response queue under link capacity constraint until the total service chunks number reached (U_p*T)/C. The pseudocode of the algorithm is described as follows:

Algorithm 1 Pseudocode for GBRA algorithm at supplier p

```
Struct ChunkRequestList{
    Size_t  receiverPeerId;
    Time deadline;
    Time sendTime;
    Time receiveTime;
} set_RequestChunks;

S: temporary array to save the request chunks which satisfy the time constraint
```

Input:

```
$U_p$: the outbound bandwidth capacity of supplier peer p
Array $U_p$[i]: the maximum bandwidth capacity of overlay link<p,i>;
T: service interval;
C: the chunk size;
set_RequestChunks: set of requested chunk sequence;
```

Output:

```
Matrix CS : the response queues for each receiver
```

GBRA-Algorithm:

```
GBRA ($U_p$, set_ RequestChunks, Array $U_p$,C,T)
CS = NULL;
CR=Null;
For j=0 to set_RequestChunks.length do
  CR = set_RequestChunks[j];
  If CR.deadline-CR.sendTime-2*(CR.receiveTime-CR.sendTime)>0 then
     ComputerPriority(CR);
     Push(S, CR);
  End If
End For
SS<—sort S according to requested chunk priority in descending order
Index = 0;
```

```
Count = 0;
qSize = 0 ;
While index < ($U_p$*T)/C do
   For j=0 to SS.length do
     Count = CS[SS[j].receiverPeerId].length;
     qSize = (Up[CS[SS[j].receiverPeerId]]*T)/C;
     if(Count < qSize) then
        push(CS[SS[j].receiverPeerId],SS[j]);
        index++;
     end If
   end For
end While
return CS
```

The computational complexity of the proposed method is decided by the uplink bandwidth of supplier and the length of the requested chunks sequence. However, since the number of requested chunks of each receiver is few during each interval, so we believe the calculation load can be neglected.

5. Performance Evaluations

In this section we carried out our simulation to evaluate the performance of the proposed scheme (Priority-based method) with FCFS method. In the FCFS scheme, the supplier side adopts first-in-first-out way to response chunks request.

5.1. Simulation Setup

We use P2PTV-sim[9] and make some extension to conduct a series of simulations to study the impact of our supplier side scheduling algorithm.

The number of peers is 500. 45% peers are DSL nodes with 400Kbps uplink bandwidth, 40% peers are Cable nodes with 800Kbps uplink bandwidth and 15% peers are Ethernet nodes with 1500Kbps uplink bandwidth. We suppose the download bandwidth of peers is infinite. The default uplink bandwidth of source node is 5Mbps. We employ real-world end-to-end latency matrix (2500*2500) measured on the Internet provided by Meridian project[10]. We use fixed random neighbor selection method to construct the overlay [14]. The number of neighbor of each peer is 20. During the simulation, the total number of delivery chunks is 200. And the receivers adopt LU/RP(Latest Useful Chunk, Random Peer)[12] scheduling algorithm.

The video rate of the encoder is a free parameter that we vary to enforce different values of the system load. The playback urgency factor α is 0.5.

5.2 Simulation Results

To estimate the system's network performance, we mainly focus on the metrics [13]: chunk delivery latency, chunk miss ratio, and peer bandwidth utilization and simulate a stable environment. While the whole session only persist less than 20s, only a marginal percentage of peers are expected to leave or join the system. So the effect of peer churning is neglected at first. And we will explicitly assess its impact in our future works. When all the nodes join in an initialization period, they persist in the lifetime of the streaming. All results are averaged at least ten independent simulation runs.

Figure 1 shows the average chunk delivery delay as a function of the target video bitrate with a certain playout delay (5s). We observe that with the increment of the video bitrate, the average chunk delivery latency increases. That's because as the video coding rate increases, the chunk size increases as well and therefore the diffusion of a given chunk takes longer. When the video rate is less than 0.5Mbps, FCFS scheme is better than priority-based scheme due to its simplicity and needn't calculate. However as the video rate grows, priority-based scheme outperforms FCFS scheme. This can be attributed to the nature of priority-based scheme which uses the more effective bandwidth to distribute the more valuable chunks and pre-filtering also benefits to remedy the consumption of calculation time. When the video rate is larger than 1.0 Mbps, the chunk delivery delay descends because a large number of chunks are lost and we use the on-time arrived chunks to compute the delay. Figure 2 shows the max chunk delivery delay as a function of the target video bitrate. With the video rate grows, the max chunk delivery delay increases and the performance of priority-based scheme is better than FCFS scheme. However they all less than 5s for we set the playout delay is 5s and chunks which miss the deadline will be lost.

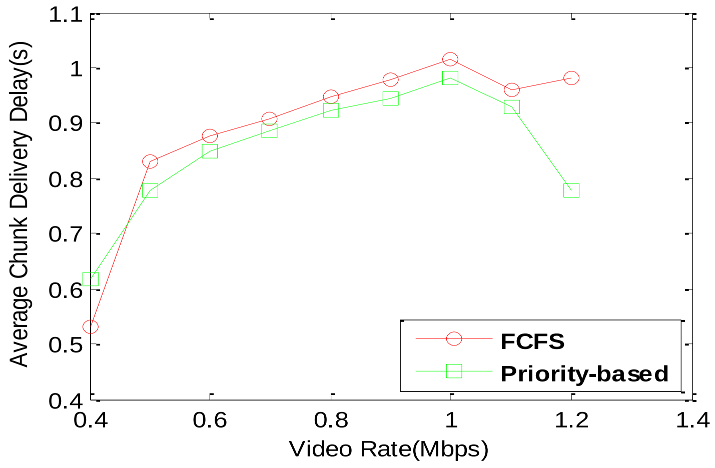

Figure 1. Average chunk delivery latency as a function of the video rate

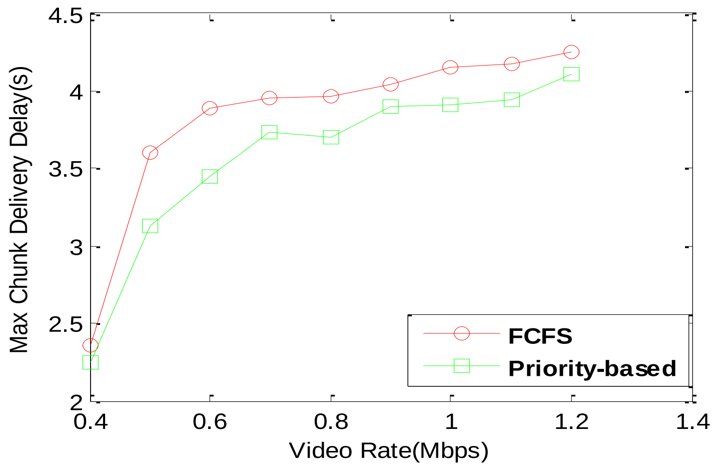

Figure 2. Max chunk delivery latency as a function of the video rate

Figure 3 shows the average chunk loss ratio as a function of the target video bitrate with a certain playout delay (5s). When a chunk is not received within its playout time, it is deemed “lost”. When the video bitrate is low and the system is under-loaded, *e.g.* video bitrate is no more than 0.6Mbps; the chunk loss rate is less than 10%. However when the video rate grows, the system’s total requirement increases while the total supply unchanged, that’s the total load increases. As a result, the chunk delivery delay becomes longer which causes the number of postponing chunk increment at a given target playout delay. So the number of lost chunks depends on the media bitrate. The loss ratio in our proposed scheme is lower than FCFS scheme.

Figure 4 depicts the average uplink bandwidth utility as a function of video rate. From Figure 7 we observe when the system’s bandwidth is rich, the uplink bandwidth utility increases as the video bitrate grows. But when the system’s requirement is larger than peers’ supply, that’s the load of the system is larger than it can support, the bandwidth utility is descending fast. Especially when the video rate is 0.9Mbps, the resource index is smaller than 0.9. That’s because with the increment of video rate, the chunk delivery delay increases which causes the number of useless chunks missing the playback deadline increase. So the ratio of the number of useful chunks received to total number of delivery chunks decreases. Our proposed method still outperforms the FCFS scheme.

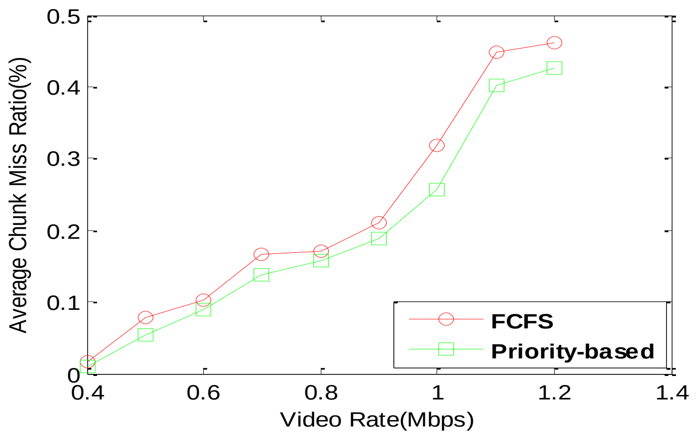

Figure 3. Average chunk miss ratio as a function of the video rate

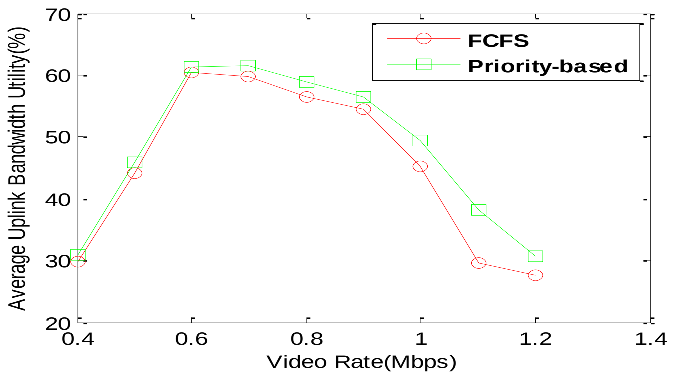

Figure 4. Average bandwidth utilization as a function of the video rate

6. Conclusion

In this paper, we study the supplier side scheduling problem in chunk-based mesh-pull P2P live streaming system. We have proposed the supplier side chunk priority model considering requested chunks' urgency of playback and rarity. Based on the model, we have formulated the supplier side scheduling problem and derived a greedy solution for it. Simulation results have shown that our scheme can achieve higher performance than techniques commonly used.

Acknowledgements

This paper is supported by China Postdoctoral Science Foundation (2013M531711), the National Natural Science Foundation of China (NO.61072051) and Technical Supporting Pro grams Funded by Ministry of Science & Technology of China (2013BAH18F02, NO.2012BA D35B02).

References

[1] L. Zhengye, S. Yanming, K. W. Ross, S. S. Panwar and Y. Wang, "LayerP2P: Using Layered Video Chunks in P2P Live Streaming", J. IEEE Trans. on Multimedia, vol. 11, no. 7, **(2009)**, pp.1340-1352.

[2] Y. Guangxue, W. Nanqing, L. Jiansheng, X. Xiaofeng and X. Linquan, "Survey on Scheduling Technologies of P2P Media Streaming", J. OF Networks, vol. 6, no. 8, **(2011).**

[3] A. Carta, M. Mellia, M. Meo and S. Traverso, "Efficient Uplink Bandwidth Utilization in P2P-TV Streaming Systems", Proceedings of the 2010 IEEE Global Telecommunications Conference (GLOBECOM 2010), (**2010)** December 6-10; Miami, Florida, USA.

[4] R. Birke, C. Kiraly, E. Leonardi, M. Mellia, M. Meo and S. Traverso, "Hose Rate Control for P2p-Tv Streaming Systems", Proceedings of the 2011 IEEE International Conference on Peer-to-Peer Computing (P2P), **(2011)** August 31-September 2; Kyoto, Japan.

[5] H. Jin and X. Tu, "Nearcast: A Locality-Aware P2P Live Streaming Approach for Distance Education", J. ACM Trans. on Internet Technology (TOIT), vol. 8, no. 2, **(2008)**.

[6] F. Pianese, D. Perino, J. Keller and E. W. Biersack, "PULSE: An Adaptive, Incentive-Based, Unstructured P2P Live Streaming System", J. IEEE Trans. on Multimedia, vol. 9, no. 8, **(2007)**, pp. 1645-1660.

[7] B. Cheng, "Research on Data Dissemination for Peer-to-Peer Video-on-Demand Systems", Ph.D Dissertation, Huazhong University of Science and Technology, **(2009)**, pp. 54-58.

[8] T. Bonald, L. Massoulie, F. Mathieu, P. Diego and L. Orange, "Epidemic Live Streaming: Optimal Performance Trade-Offs", Proceedings of the 2008 International Conference on Measurement & Modeling of Computer Systems (Sigmetrics'08), **(2008)** June 2-6; Annapolis, Maryland, USA.

[9] NAPA-WINE Project, http://www.napa-wine.eu/cgi-bin/twiki/view/Public.

[10] Meridian-Project, http://www.cs.cornell.edu/People/egs/meridian/data.php.

[11] R. Kumar, L. Yong and K. Ross, "Stochastic Fluid Theory for P2P Streaming Systems", Proceedings of the 26th IEEE International Conference on Computer Communications, **(2007)** May 6-12; Anchorage, Alaska.

[12] Y. Hongyun, C. Xuhui and H. Ruiming, "An End-to-End Congestion Control Approach based-on Content-Aware", International Journal of Multimedia and Ubiquitous Engineering, vol. 4, no. 2, **(2009)**, pp. 81-90.

[13] Y. Hongyun, H. Ruimin and C. Xuhui, "Contribution and QoS-Aware Neighbor Selection for Peer-to-Peer streaming", International Journal of Multimedia and Ubiquitous Engineering, vol. 8, no. 2, **(2013),** pp. 123-142.

Authors

Hongyun Yang

She received the Ph.D. degree of electronics and communication from Wuhan University, Wuhan, China in 2012. Now she is a post doctor at National Engineering Research Center for E-Learning, Central China Normal University. She was a student Member of IEEE, ACM and CCF. Her research interests include cloud computing, Dynamic Adaptive Streaming based-on Http (DASH), P2P streaming and distance education.

Xiaoliang Zhu

He is an associate professor at National Engineering Research Center for E-Learning, Central China Normal University. He received the Ph.D. degree from Huazhong University of Science and Technology,Wuhan,China in 2006.His research interests include Distance Education, Multimedia Information Processing and Multimedia Communications

Xuhui Chen

He is a lecturer at College of Mathematics & Computer Science, Wuhan Textile University. He received the Ph.D. degree of Computer Architecture from Wuhan University, Wuhan, China in 2010. He was a Member of IEEE. His research interests include Embedded System, Computer Architecture and Multimedia Communication.