2010 10th IEEE International Conference on Computer and Information Technology (CIT 2010)

Dynamic Resource Provision in Multi-Channel P2P Live Streaming Systems

Xiaoqun Yuan
Department of Electronics and Information Engineering
Huazhong University of Science Technology
Wuhan, 430074, China
Email: yuanxiaoqun@smail.hust.edu.cn

Hao Yin,Xuening Liu and Changlai Du
Department of Computer Science and Technology
Tsinghua University
Beijing, 100084, China
Email: { h-yin, liuxn99, duchanglai } @ mails.tsinghua.edu.cn

Geyong Min
Department of Computing
University of Bradford
Bradford, BD7 1DP U.K.
g.min@brad.ac.uk

***Abstract*—Peer-to-Peer is one of the most popular live streaming delivery technologies owing to its scalability and low-cost. Most practical P2P streaming systems adopt hundreds of channels to deliver streaming to thousands of users simultaneously, and thus there is a great challenge to allocate server resources among these channels efficiently. In practice, most P2P systems have resorted to over-allocating resources of servers to different channels, which may result in low-efficiency of servers and high-cost of system operation. Unfortunately, there are few studies addressing this problem. In this paper, we focus on allocation of server resources to different channels appropriately. We first analyze the scalability of P2P systems and present the streaming quality model of P2P live streaming systems, which discloses the relation between the streaming quality of channel and corresponding influencing factors. A dynamic provisioning algorithm is then proposed for multi-channel P2P live streaming systems based on this quality model. The experimental results demonstrate that the algorithm can effectively improve the channel streaming quality for multi-channel P2P live streaming systems.**

I. INTRODUCTION

With the emergence of live streaming business, the Internet has recently experienced a significant increase in these applications, and some live streaming distribution technologies were developed to apply them, among which P2P is one of the most popular technologies. In fact, most multi-channel P2P live streaming systems, such as Coolstreaming[1], PPLive, UUSee and livesky [2], broadcast live multimedia content to millions of users simultaneously through hundreds of media channels. Unfortunately, there always exists the problem that the streaming quality is unsatisfactory. In order to solve this problem, most P2P systems have resorted to the practice of over-provisioning server capacities to satisfy the streaming demand from peers in each of its channels, which means more cost and lower efficiency.

It is a great challenge for P2P live streaming systems to allocate appropriate server bandwidth resource to different channels. However, most existing studies focus on single channel P2P streaming systems, such as constructing more effective P2P topology[9][10], copping with peer churn and maintaining the quality of live streams[11][12][13] et al, but there are few works focusing on multi-channel P2P systems till 2008. C.Wu [3] analyzed 7 months of run-time traces data from UUSee, then proposed an online server resource provisioning algorithm for each channel, but the model was just based on the trace data and didn't analyze general multi-channel P2P live streaming system. D. Wu [4][5] proposed infinite-server queuing network models of the multi-channel P2P streaming system, then applied the models to the isolated channel (ISO) and the View-Upload Decoupling (VUD) P2P streaming designs. Obviously, while this model ia based on the infinite queuing model and is suitable for P2P streaming system designs, it is not suitable for online resource provision among hundreds of channels due to the dynamic P2P. So it is necessary to develop a general dynamic provisioning algorithm for multi-channel P2P live streaming systems.

Obviously, the core goal of this provisioning algorithm is to maximize the P2P resource utility, and is similar to the traditional network resource allocation problem, which has been well studied. In last two decade, there are many network resource allocating algorithms have been proposed [6][7][8]. Most of these allocating algorithms focused on the adaptation of source rates and routing to efficiently use network resources, and maximized the network resource utility of Layer three(TCP) or Layer Four(IP). Due to the unique feature of peer acting as both receiver and sender and the feature of overlay network, these allocating algorithms are not suitable for P2P systems and a new server resource allocating algorithm needs to be developed to capture the essential aspects of P2P multi-channel systems.

In this paper, we try to solve the problem of the server resource allocation among different channel. In order to do this, we first analyze the relationship between the demanded capacity of P2P system and the server upload capacity, then we present a general streaming quality model. Based on this model, we proposes a dynamic resource allocating algorithm for multi-channel P2P live streaming systems. Our contribution s are summarized as fellows:

978-0-7695-4108-2/10 $26.00 © 2010 IEEE
DOI 10.1109/CIT.2010.318

1) We present a general streaming quality model of multi-channel P2P live streaming systems, which discloses the relationship between the streaming quality of channel and the factors, such as peer number, server upload capacity and the average peer contribution of each channel.

2) Based on the the streaming quality model, we proposes a dynamic resource allocating algorithm for multi-channel P2P live streaming systems, which allocating server resources among different channels based on the corresponding peers demand of each channel of the systems and can improve the streaming quality of the P2P systems.

The rest of this paper is organized as follow. In section II, we present our basic system model of multi-channel P2P live streaming system, including some basic assumptions and notations, and formulate the problem of the server resource allocating among channels as a optimal allocating problem. In sections III, we present the detailed analysis of the resource demand and the streaming quality model of multi-channel P2P live streaming systems. And in section IV, based on the streaming quality model, a practical dynamic server resource allocating algorithm is proposed and the validity of this algorithm is confirmed by our simulation. In section V, we summarize this paper and discuss the further work.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. System Overview

In this section, we present our basic multi-channel P2P live streaming system model, including the underlying assumptions and notations (Table I).

Our P2P live streaming system has three major components: 1) the server, which manages and provides live streaming for our P2P system; 2) channels, through which servers broadcast streaming to users, and through each channel servers broadcast the same streaming to corresponding peers; and 3) users, which not only download streaming from the server and other peers but also upload streaming to other peers.

In this system, all peers are divided into some subsystems according to programming content, such as Sports subsystem for live sports program, News subsystem for the News all around the world, and for each subsystem k, there is a corresponding channel k through which the server broadcasting streaming to peers. Peers in subsystem k are organized as a mesh-based P2P topology. At each time, the system simultaneously accommodates users with many channels of programming and the server allocates bandwidth resource to these different channels to service peers and to satisfy the streaming demand from peers. In the system, different subsystems are independent each other and each subsystem has no information redundancy. That is to say, each peer just download streaming from servers and the other peers in the same subsystem, and the departure or join of peers in one subsystem don't affect any other subsystem and in each subsystem, all content uploaded by peers are downloaded by peers in the same subsystem. In this system, the time of peer switch is negligible. When a peer leaves, the neighbor peers can download the data form the other alive peers at once, and when a peer joins, it can download(upload) streaming from(to) other peers.

Table I
NOTATIONS USED IN OUR PAPER

Notation	Definition
U	The total bandwidth capacity of the server.
M	The total channel number through which the server broadcasts streaming to peers.
r_s^k	The bandwidth capacity which the server service peers through channel k , r_j^k expresses the bandwidth capacity which peer j downloads from all the other peers and r^k denotes the total download capacity of all peers in subsystem k.
b_i^k	The total capacity which peer i uploads to other peers in subsystem k.
ρ^k	Average fraction contribution capacity of peers in channel k
R^k	Live streaming rate of channel k.
Q^k	Expected quality of the channel k and q_i^k denotes the quality of peer i.

To build our quality model, which can reveal essential aspects of practical system, yet be still simple enough, we make the following assumptions:

1) In each channel k, users are organized as a mesh-based P2P topology, all peers in the same channel have the Independently and Identically Distribution(IID). In each time slot, the change rate of peer number of channel k is expressed as p^k, and each new peer will randomly select m partners from the current set of alive peers to ask for their upload capacity.

2) The streaming quality refers the ratio of peer download capacity to streaming rate R^k, and the high quality means that the peer's download capacity is no less than streaming rate R^k. At the time t, the high quality probability of channel k is β^k.

B. Problem Formulation

The problem of multi-channel resource allocation can be defined as follow:

Considering a multi-channel P2P system with a server and M channels, let U denote the total upload capacity of server, R^k expresse the streaming rate of channel k, N^k present the peer number of subsystem k, and p^k denote the change rate of peer's number of channel. At this condition, the multi-channel resource allocation can be described as determining the optimal allocation server upload bandwidth capacity among different channels. Obviously, it is an optimal allocating problem with multiple constraints:

Maximize:

$$\sum_{i=1}^{M}(Q^k)^a(\delta^k)^b \tag{1}$$

Subject to:

$$\mathrm{U} \geq \sum_{i=1}^{M}(r_s^k) \tag{2}$$

$$\mathrm{Q^k} = f(N^k, r_s^k, \rho^k) \tag{3}$$

where δ^k denotes the predictability of the streaming quality of subsystem k, Constraint(2) denotes that the total allocating capacity is less than the server upload capacity and Constraint (3) shows that the streaming quality of channel k is the function of N^k, r_s^k, ρ^k . In this model, we uses δ^k as weight parameter to control server capacity allocation, which assures some subsystems to achieve relative good streaming quality during some certain period, and to avoid the universal bad streaming quality.

III. P2P SCALABILITY MODEL

A. P2P Scalability Model

Considering channel k, let the peer number of channel k be N^k, r_{si}^k denote the download capacity of peer i from server, r_i^k express the total download capacity of peer i from other peers, and b_i^k present the total upload capacity of peer i to others peers. Then in subsystem k, the relationship between the download capacity and upload capacity of peer i can be described as:

$$(\mathrm{r_{si}^k} + \mathrm{r_i^k})\rho^\mathrm{k} = b_i^k \tag{4}$$

Let N denote the peers' number of channel k, then we have

$$\sum_{\mathrm{i=1}}^{\mathrm{N}}(\mathrm{r_{si}^k} + \mathrm{r_i^k})\rho^\mathrm{k} = \sum_{i=1}^{N} b_i^k \tag{5}$$

where $\sum_{i=1}^{N} r_{si}^k = r_s^k$, denoting the capacity allocated by the server, $\sum_{i=1}^{N} b_i^k = b^k$, denoting the capacity uploaded by peers, and $\sum_{i=1}^{N} r_i^k = r^k$, denoting the capacity which peers download form other peers.

Since different channels have no information exchange and there is no information redundancy in each channel. Obviously, in subsystem k, $r^k = b^k$, so the P2P scalability model can be described as:

$$(\mathrm{r_s^k} + \mathrm{r^k})\rho^\mathrm{k} = r^k \tag{6}$$

where r_s^k is the total capacity that the server allocates to channel k, and r^k is the total contribution of peers in subsystem k.

Equation (6) shows that the scalability of channel k and the contribution of peers in channel k are the function of r_s^k and ρ^k. In the case of $\rho^k \geq 1$, if equation (6) holds, then $r_s^k \leq 0$. This means that the peers of channel k upload capacity to server through channel k, which is in conflict with the fact that the server broadcasts live streaming to peers in channel k. So when $\rho^k \geq 1$ and $r_s^k \geq R^k$, the peers of channel k not only provide enough capacity for all peers to view live streaming smoothly, but also provide some surplus upload capacity for further scaling. What's more, with the population of peers increasing, peers in channel k have a better streaming quality and the channel k has more scalability. But in the case of $\rho^k < 1$, if equation (6) holds, then $r_s^k > 0$. This means that the server needs to provision some demanded capacity to maintain channel quality. In this case, if the server uploads the same capacity, the streaming quality of channel k decreases with the peer number increasing.

B. Channel Quality

The P2P scalability model (6) shows that when $\rho^k \geq 1$, the server only provides streaming of $r_s^k(r_s^k \geq R^k)$ to achieve high streaming quality, but when $\rho^k \leq 1$, it need upload more demanded capacity to provide good streaming quality. In this paper, we will focus on the latter.

For subsystem k, let β^k denote the peer's high download probability, which means that the download capacity of peer i is more than R^k, and let v_i^k express the viewing rate of peer i. Then we formulate the viewing rate of peer i as follow:

$$v_i^k = \begin{cases} R^k & r_{si}^k + r_i^k \geq R^k \\ r_{si}^k + r_i^k & r_{si}^k + r_i^k < R^k \end{cases} \tag{7}$$

We define the streaming quality q_i^k of peer i as $q_i^k = E[P\{v_i^k : v_i^k \geq R^k\}]$. Substituting Equation (7) into it, the q_i^k can be further described as:

$$q_i^k = \begin{cases} 1 & r_{si}^k + r_i^k \geq R^k \\ \beta^k + (1-\beta^k)\frac{r_{si}^k + r_i^k}{R^k} & r_{si}^k + r_i^k < R^k \end{cases} \tag{8}$$

Considering channel k, let q_i^k express the expected probability of user viewing rate. Then we have:

$$\mathrm{q_i^k} = \beta_i^k + \frac{(1+\beta_i^k)(r_{si}^k + r_i^k)}{R^k}, r_{si}^k + r_i^k \leq R^k \tag{9}$$

In channel k, all peers have the Independently and Identically Distribution (IID), so the peers' expected average streaming quality of channel k(channel quality) can be described as fellows:

$$Q^k = \frac{\sum_{i=1}^{N_i^k} q_i^k}{N^k} \tag{10}$$

$$= \beta^k + \frac{(1+\beta^k)r_s^k}{N^k R^k (1-\rho^k)} \tag{11}$$

Equation (11) shows the relationship between channel streaming quality and the corresponding influence factors r_s^k, ρ^k and N^k.

Fig.1 shows the relationship between the streaming quality of P2P subsystem k and corresponding influence factors r_s^k, ρ^k and N^k. Fig.1(a) plots the relationship between the streaming quality of P2P systems and N^k with different server capacities, Fig.1(b) expresses the relationship between the streaming quality of P2P systems and r_s^k with different peers contributions and Fig.1(c) shows the relationship between the streaming quality of P2P systems and ρ^k with different server capacities. In Fig.1(a), the range of N^k is from 1000 to 2000, the step is 1 and the server capacity. Fig.1(a) shows that the streaming quality is decreasing with N^k increasing when the server capacity is given. Similarly, Fig.1(b) shows that the streaming quality is increasing with server capacity increasing when peer contribution is given and increasing with the peer contribution increasing. Obviously, form Fig.1, we can tell that the channel quality has a negative correlation with N^k, and has a positive correlation with ρ^k and r_s^k.

IV. A RESOURCE PROVISIONING ALGORITHM FOR MULTI-CHANNEL P2P LIVE STREAMING SYSTEMS

In this section, we present a dynamic server capacity provisioning algorithm, which probatively adjusts the server capacity available to each of the concurrent channels. Our provisioning algorithm takes into account the predictability, the number of peers and the streaming quality of each channel.

A. Parameters Estimating and Predicting

Before describing the provisioning algorithm, we firstly determine the parameters of our algorithm.

1) Estimating Peer Number: Equation (11) shows that the streaming quality of subsystem k is the function of ρ^k, r_s^k and N^k. In order to probatively adjust the server capacity for each channel, we first estimate the number of active peers in channel k at the time $t+1, N_{t+1}^k, 0 < k < M$.

At time t, assume the peer number of channel k is N_t^k. Then, at time $t+1$, we estimate the peer number of channel k through the following equation :

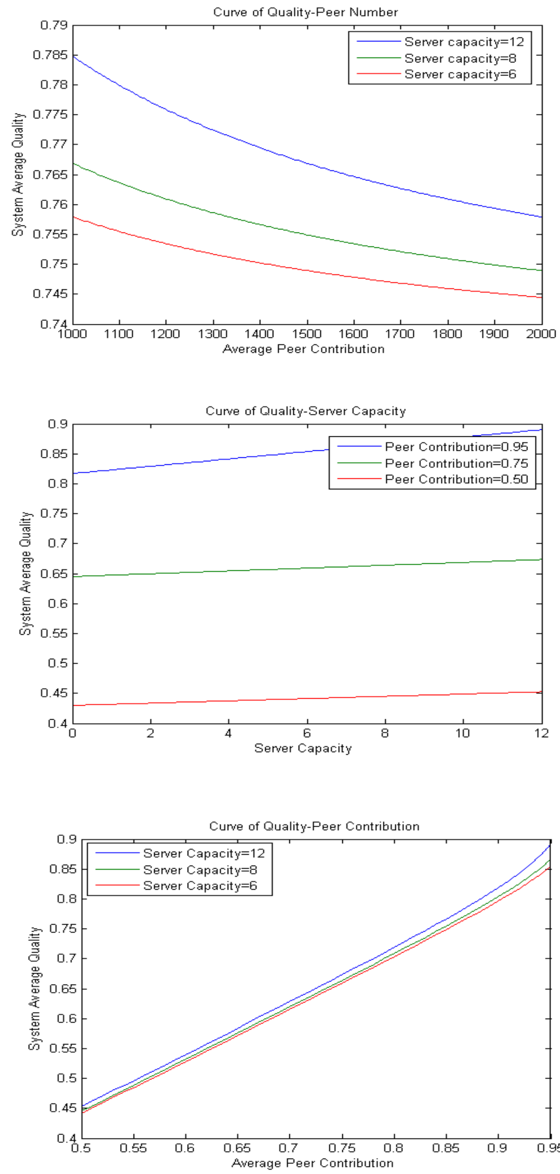
$$\overline{\mathrm{N}^{\mathrm{k}}_{\mathrm{t+1}}} = (1+p_t^k)N_t^k - \mu(N_t^k - \overline{N_t^k}) \tag{12}$$

where p_t^k is the peer number change rate of subsystem k at time t, μ is the correction parameter and $\overline{N_t^k}$ is the estimated value of peer number at time t.

2) the Prediction of Channel Quality: We estimate the probability of high streaming quality of subsystem k with $\beta_{t+1}^k = \frac{n_t^k}{N_t^k}$, where n_t^k is the peer number whose download capacity is more than R^k at time t.

Let $r_{s(t+1)}^k$ denote the capacity which server will allocate to channel k at time $t+1$, then substituting (12) and $r_{s(t+1)}^k$ into (11), the average streaming quality of subsystem k at time $t+1$ can be described as fellow:

$$\overline{\mathrm{Q}^{\mathrm{k}}_{\mathrm{t+1}}} = \beta_{t+1}^k + \frac{(1+\beta_{t+1}^k)r_{s(t+1)}^k}{\overline{N_{t+1}^k}(1-\rho^k)R^k} \tag{13}$$



(c) Vs.ρ

Figure 1. Correlation between average streaming quality and its influencing factors

B. The controllability of channel quality

In multi-channel P2P live streaming systems, all channels not only compete but also share the same server capacity. Equation (13) shows that the average quality of channel k decreases with the peer population increasing when $\rho^k < 1$, which means the over capacity allocation strategy aren't suitable at peak time. What's more, due to the decentralized ,uncoordinated operation, P2P systems have low controllability of channel service quality with undesirable disruptions.

In P2P live streaming system, the churn of peers is the most critical factor which can lead to the quantity changing greatly, and there are two kind of churns on different time scales: peers entering and leaving the system on long time scale or on short scale. The former occurs on on-device peers entering and leaving the streaming application for a long time and the latter occurs on peers joining and leaving the streaming application randomly. Since the on-device peers are always online and can provide stable upload and download capacity for a long time, which makes it possible to predict and control the streaming quality of channel for future. For example, the Set-Top-Boxes are always on-devices in one channel and have stable download and upload capacity[7]. But the random peers enter and leave streaming application randomly, so they can't service the others peer stably, which make it challenge to predict and control the channel streaming quality, especially at the peak time. Fortunately, the two kind of peers are in the same P2P system. So, even at peak time, it is possible to make some prediction of streaming quality and provide good quality streaming to peers. In this paper, we use δ_t^k to express the controllability of streaming quality of channel k at time t, and define δ_t^k as fellow:

$$\delta_{\mathrm{t}}^{\mathrm{k}} = Q_t^k / \overline{Q_t^k} \tag{14}$$

where $Q_t^k = \frac{\sum_{i=1}^{N_t^k} q_{i(t)}^k}{N_t^k}, q_{i(t)}^k = \min(1, r_{si(t)}^k + r_{i(t)}^k)$ is the actual streaming quality of subsystem k at time t, $\overline{Q_t^k}$ denotes the expected streaming quality of subsystem k at time $t-1$.

Obviously, the predictability of channel k increases with δ_t^k increasing. $\delta_t^k = 1$ means the streaming quality of channel k can be predictable and controllable completely, and $\delta_t^k \geq 1$ means the streaming quality is better than expected.

C. Optimal Allocation of Server Capacity

Based on the predicted peer number, the streaming quality and the predictability of streaming quality of each subsystem, we reformulate the model of (1) into:

Maximize:

$$\mathrm{RA} = \sum_{i=1}^{M} (Q_{t+1}^k)^a (\delta_t^k)^b \tag{15}$$

Subject to:

$$0 \leq r_{s(t+1)}^k \leq B_{s(t+1)}^k \tag{16}$$

$$\mathrm{U} \geq \sum_{i=1}^{M} r_{s(t+1)}^k \tag{17}$$

$$\mathrm{a+b} = 2, 1 \leq a \leq 2, 0 \leq b \leq 1 \tag{18}$$

$$\mathrm{Q_{t+1}^k} = f(N_{t+1}^k, r_{s(t+1)}^k, \rho^k) \tag{19}$$

where

$$\mathrm{RA} = \sum_{\mathrm{k=1}}^{\mathrm{M}} (\beta_{\mathrm{t+1}}^{\mathrm{k}} + \frac{(1-\beta_{\mathrm{t+1}}^{\mathrm{k}})\mathrm{r_{s(t+1)}}}{(1-\rho^{\mathrm{k}})\mathrm{R^k}\overline{\mathrm{N_{t+1}^k}}})^{\mathrm{a}} (\delta_{\mathrm{t}}^{\mathrm{k}})^{\mathrm{b}},$$

is the objective function. $B_{t+1}^k = (1-\rho^k)R^k\overline{N_{t+1}^k}$ is the maximal server capacity requirement for channel k at time $t+1$, in which case the channel quality achieves 1. Constraint (16) means the capacity server assigns to channel k is no more than the maximal server capacity requirement of channel, constraint (17) denotes that the total capacity assigned to channels is subject to the service ability of server.

In order to find an optimal solution to the optimization problem, we introduce Lagrange multiplier operator[8]. Let Lagrange multiplier λ for the constraint in (21), μ for the constraint in (22), υ for the constraints in (23),and ω for the constraint in (24). Then we get the KKT conditions for this allocation problem as follows:

$$-\frac{\partial RA}{\partial r_{s(t+1)}k} + \lambda + \mu + \nu + = 0 \tag{20}$$

$$\lambda(\mathrm{U} - \sum_{i=1}^{M} r_{s(t+1)}^k) = 0 \tag{21}$$

$$\mu(r_{s(t+1)}^k - B_{s(t+1)}^k) = 0 \tag{22}$$

$$\upsilon r_{s(t+1)}^k = 0 \tag{23}$$

$$\omega(a+b-2) = 0 \tag{24}$$

For $(r_{s(t+1)}^k)^* \geq 0$ and $(r_{s(t+1)}^k)^* \geq B_{r(t+1)}^k$, then from constraints (22) and (23), we $(\mu)^* = 0$ and $\nu^* = 0$. Substituting it into equation (21), we have $\frac{dRA}{dr_{s(t+1)}^k} = \lambda^*$. Introducing a new equation $\pi = \frac{1-\beta_{t+1}^k}{(1-\rho^k)R^k\overline{N_{t+1}^k}}$. From equation (15), we have $\frac{dRA}{dr_{s(t+1)}^k} = \frac{\lambda^{\frac{1}{a-1}}}{\pi^{\frac{a}{a-1}}c^{\frac{1}{a-1}}} - \frac{\beta_{t+1}^k}{\pi}$. Finally, we derive the optimal solution as fellow:

$$r_{s(t+1)}^{k*} = \begin{cases} \beta_{s(t+1)}^k & \lambda^* < c\pi(\beta_{s(t+1)}^k + \pi\beta_{s(t+1)}^k) \\ \frac{\lambda^{\frac{1}{a-1}}}{\pi^{\frac{a}{a-1}}c^{\frac{1}{a-1}}} - \frac{\beta_{t+1}^k}{\pi} & \lambda^* \geq c\pi(\beta_{s(t+1)}^k + \pi\beta_{s(t+1)}^k) \end{cases} \tag{25}$$

where $c = a-1$, which controls the participation of δ_t^k in server resource provision. The optimal server bandwidth provisioning for channel k, $r_{s(t+1)}^{k*}$,$0 < k \leq M$, can be obtained with a water-filling approach.

D. Dynamic Server Capacity Allocation

In our algorithm, the server capacity allocation is periodically carried out to adapt to the changing capacity demand of each channel. In order to minimize the computation overhead, we propose an incremental water-filling approach, which can adjust server capacity shared among the channels from their previous values.

At time t, our approach starts with $r_{s(t+1)}^k = (r_{s(t)}^k)^*$. We first collect the data from peers of each channel, then compute the correlative parameters of (15) at time t and some prediction parameters for time $t+1$. After that,

Table II
THE DYNAMIC SERVER CAPACITY PROVISIONING ALGORITHM

At time t, the designated server

1. Peer information collection

Collect the number of active peers in each channel,N_t^k ,and each peer's state information, including of r_i^k and β_i^k , then derive the streaming quality, the average streaming quality of system and the quality controllability for each channel.

2. Channel parameters prediction

Predict the peer population, the average fraction contribution capacity, the average streaming quality of each channel, and the expect quality of each channel for the time $t+1$.

3. Proactive server capacity provisioning for all the channels.

1) Compute current surplus of server capacity with $U - sum_{k=1}^{M} r_{s(t+1)}^k$ and $r_{s(t+1)}^k - B_{s(t+1)}^k$.

2) Allocate surplus to channels with$B_{s(t+1)}^k - r_{s(t+1)}^k$.

3) Adjust channel capacity assignment with incremental water-filling approach.

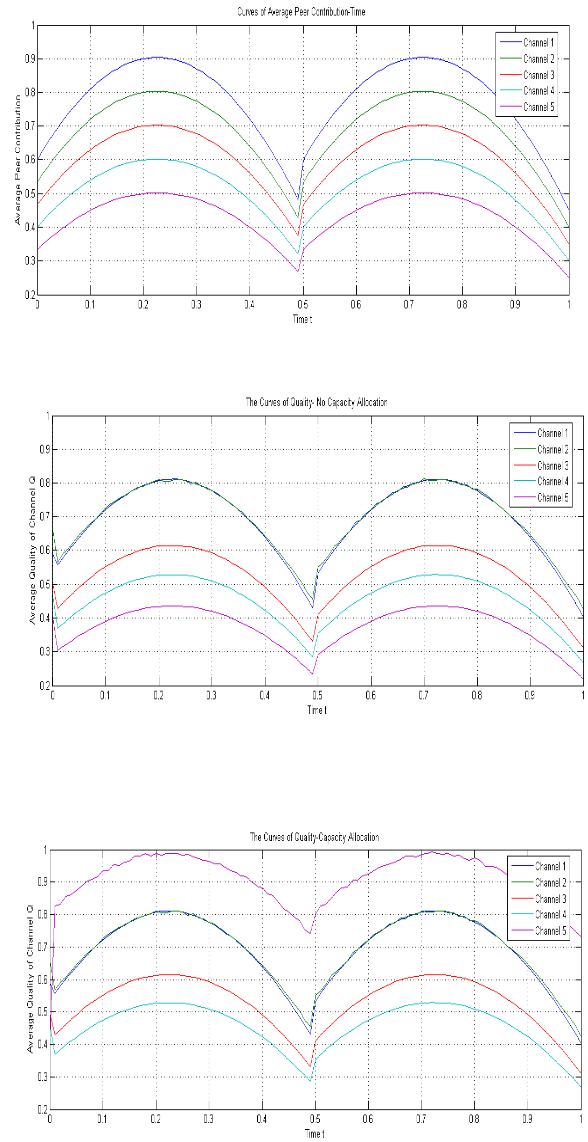
---

we pre-allocate server capacity provision for all channels, which includes three process: 1) collecting surplus capacity, which collects any surplus of the overall provisioned server capacity for all channels at time t; 2) provisioning surpluses capacity, which allocates the surpluses capacity to the channels whose maximal server capacity requirement has not been reached; and 3) adjusting the server capacity assignment towards the achievement of a same marginal utility across the channels, which repeats until all channels have reached the same marginal utility, or have reached their respective maximum server bandwidth requirement. Our complete algorithm is summarized in Table II, which is periodically carried out on a designated server. Each peer in channel k is asked to periodically send its state information, such as download capacity, upload capacity, to the server.

E. Experience Design and Performance Analysis

We develop a set of Matlab programs to calculate the performance of our algorithm. In order to ensure our model revealing the real Internet environment, we let ρ^k denote the real Internet environment traffic, which is generally better at early hours of a day, degrades to a low value around the noon time, then increases slightly, and drops to its daily lowest value before midnight, shown as figure 2-a. In order to signify the randomness of peers in P2P system, we add some churn to the number calculated with equation (15) at each time slot. In the default settings, there is a random number within 2000, which denotes peer population in the system, 5 channels, the respected rate of channel is 0.4, 0.3, 0.43, 0.51 and 0.37 and the respected capacity of channel is 13, 17, 12, 10, 8.

Fig.2(a) shows the change curves of each channel quality with time varying. Fig.2(b) shows the change curves of each channel quality for the no capacity adjusting case and Fig.2(c) shows the change curves of each channel quality for the dynamic adjusting capacity with our algorithm. Comparing Fig.2(b) to Fig.2(c), we find that our source



(c) Vs.Dynamic server capacity allocation

Figure 2. Streaming quality achieved with different scenarios

allocating algorithm improves the quality of both the system and each channel. But Fig.2(b) and Fig.2(c) tell us that the improvement of channel quality is not significant. There are two reasons: one is that we limit the value of random noise which denotes the churn of peer number change within 20, comparing 2000 at beginning; the other is that we assign control parameterμ with small value, which further decreases the impact of peer churn.

V. CONCLUSION

In this paper, we study the channel streaming quality of multi-channel P2P live streaming systems using a close quality model. Our model captures the essential aspects of multi-channel P2P system. The analytic model makes it possible to assign appropriate server resource to different channel to satisfy different applications. Then we propose a dynamic server resource provisioning algorithm based on our quality model.

Although our experience verifies that our algorithm can effectively improve the streaming quality of channel for multi-channel P2P systems, our algorithm should be deployed into practical P2P system to confirm its value.

REFERENCES

[1] X.Zhang, J.Liu, B.Li and T.-S.P.Yum, *DONet/CoolStreaming: A Data-driven Overlay Network for Peer-to-Peer Live Media Streaming*, in Proceedings of IEEE INFOCOM, Mar. 2005, Vol.3, pp. 2102-2111, 2005.

[2] H.Yin, X.Liu, V.Sekar, F.Qiu, C.Lin, H.Zhang and B.Li, *Design and Deployment of a Hybird CDN-P2P System for Live Video Streaming: Experiences with LiveSky*, In Proceedings of the seventeen ACM international conference on Multimedia 2009, pp. 25-34, 2009.

[3] C.Wu, B.Li and S.Zhao, *Multi-channel Live P2P Streaming: Refocusing on Servers*, in Proceedings of IEEE INFOCOM 2008, pp. 1355-1363, 2008.

[4] D.Wu, Y.Liu and K.Ross, *Queuing Network Models for Multi-Channel P2P Live Streaming Systems*, in the Proceedings of IEEE INFOCOM 2009, pp. 73-81. 2009.

[5] D.Wu, C.Liang, Y.Liu and K.Ross, *View-Upload Decoupling: A Redesign of Multi-Channel P2P Video Systems*,in the Proceedings of IEEE INFOCOM Mini-Conference 2009, pp. 2726-2730 2009.

[6] M.Cha, P.Rodriguez, S.Moon and J.Crowcroft, *on next-generation telco-managed p2p tv architectures*, in the Proceedings of IPTPS 2008 .

[7] S.Boyd and L.Vandenberghe, *Convex Optimization*, Cambridge University Press, New York, America, 2004.

[8] M.Castro, P.Druschel, A.-M.Kermarrec, A.Nandi, A.Rowstron and A.Singh, *SplitStream: Highbandwidth content distribution in a cooperative environment*, in Proc. 2nd International Workshop on Peer-to-Peer Systems, 2003.

[9] D. Kostic, A. Rodriguez, J. Albrecht and A. Vahdat, *Bullet: high bandwidth data dissemination using an overlay mesh*, in Proceedingsof 19th ACM Symposium on Operating Systems Principles, Vol. 37, No. 5, pp. 282-297, October 2003.

[10] T.Small, B.Li and B.Liang, *On Optimal Peer-to-Peer Topology Construction with Maximum Peer Bandwidth Contributions*,in the Proceedings of the 23rd Queen's Biennial Symposium on Communications, pp.157-160, 2006.

[11] B.Cohen, *Incentives Build Robustness in BitTorrent*, In Proceedings of the Ist Workshop on Economics of P2P Systems Cambridge University Press, New York, America, 2004.

[12] X.Yang and G.de Veciana, *Service Capacity of Peer to Peer Networks*, In Proceedings of IEEE INFOCOM 2004 Vol. 4, pp. 2242-2252, 2004.

[13] D.Rubenstein and S.Sahu, *Can unstructured p2p protocols survive flash crowds?*, IEEE Transactions on Networking, vol.13, no.3, pp.501C512, Jun. 2005