Scalable and Reliable Live Streaming Service through Coordinating CDN and P2P

[1]ZhiHui Lu
[1,3,4]School of Computer Science, Fudan University
Shanghai, China, 200433
[1]lzh, [3]082024070, [4]082024093@fudan.edu.cn

[2]XiaoHong Gao, [3]SiJia Huang, [4]Yi Huang
[2]Department of Media Communication,
Donghua University,Shanghai, China, 201620
[2]gaoxiaohong@dhu.edu.cn

***Abstract*-In order to fully utilize CDN network edge nearby transmission capability and P2P scalable end-to-end transmission capability , while overcoming CDN's limited service capacity and P2P's dynamics, we need to combine the CDN and P2P technologies together. In recent years, some researchers have begun to focus on CDN-P2P-hybrid architecture and ISP-friendly P2P content delivery technology. In this paper, we firstly make an analysis on main problems of CDN-P2P-hybrid technology, and we compare some current existing solutions. And then we propose a novel scalable and reliable live streaming service scheme through coordinating CDN and P2P. In this scheme, different overlay hybrid methods, we design a smoother CDN and P2P overlay hybrid approach. After that we design a new peer node buffer structure for consuming media block from both CDN and P2P sources. And then we propose a novel algorithm for identifying and serving super node. The simulation experiment results show that our CDN-P2P-hybrid based live streaming scheme has better performance and reliability than pure P2P. Finally, we make a conclusion and analyze future research directions.**

Keywords- CDN; P2P; P2P Streaming; CDN-P2P-hybrid Architecture; Live Streaming; VoD Streaming

Note: This work is supported by 2009 National Science Foundation of China (60903164): Research on Model and Algorithm of New-Generation Controllable, Trustworthy, Network-friendly CDN-P2P hybrid Content Delivery.

I. INTRODUCTION

In recent years, Internet Protocol Television (IPTV), Web-based Internet streaming video, and high-definition video have become popular broadband streaming applications. The high-bandwidth, high-traffic and high QoS requirements of these applications, have brought huge challenges to the current best-effort Internet. How to implement reliable, scalable, low cost, and QoS guaranteed streaming service has become one core problem.

CDN and P2P are two mainstream technologies to provide streaming content services in the current Internet. CDN and P2P have their own advantages, but constrained by the service model, they both have some fundamental shortcomings. CDN can provide reliable and stable service, but its service capability is limited and difficult to extend; while P2P service is extensible and scalable, but P2P service is dynamic and unstable. Therefore, it is obvious that the two service models are compatible. In Service-Oriented Architecture (SOA), composing multiple services, rather than accessing a single service, is essential and provides more benefits to users. Similar with SOA, for the benefits of backbone network carriers and users, combining the P2P service and CDN service, will best protect the previous investment and improve the service capability at the same time. Indeed, with recent rapid growth of P2P applications and CDNs, many industrial and academic initiatives have been taken to combine the two technologies to get the "best of both worlds", which means obtaining the scalability advantage of P2P, as well as the reliability and controllability advantages of CDN. Some researchers have begun to focus on ISP-friendly, reliable P2P content delivery technology (e.g., [4, 7, 9-11]) and CDN-P2P-hybrid architecture (e.g., [12-22,29]).

We find that CDN P2P hybrid key points are how to mix their overlay network, and how to dynamically allocate their respective content service shares. Therefore, there are two key issues arising in CDN-P2P-hybrid architecture: one is the overlay network hybrid issue; another is the peer node playing buffer hybrid issue. The first issue focuses on deciding where to download the content, and the second issue focuses on deciding which content blocks to download from the selected CDN or P2P nodes.

In this paper, we firstly explore and analyze current solutions for overlay hybrid issues. After that, we propose a novel CDN-P2P-hybrid live streaming service system to solve these two issues.

To summarize, the key contributions and key findings of this paper are as follows:

We conduct a thorough analysis on how the existing CDN-P2P hybrid schemes deal with the major issues and problems arising in the CDN and P2P integration process, especially CDN-P2P overlay network hybrid issues.

Based on the analysis, we design a novel CDN-P2P-hybrid service scheme to realize controllable, scalable, and reliable live streaming service system. We present key algorithms and formulations to coordinate CDN and P2P, and we address some key challenges in peer node buffer structure and super node identifying and serving algorithm.

The simulation experiments have validated that this new architecture has better performance and better reliability than the pure P2P streaming system.

The remainder of this paper is organized as follows: in Sec.II, we explore and analyze the overlay issues of CDN-P2P-hybrid delivery and compare some existing solutions for these issues. In Sec. III, we design a novel CDN-P2P-hybrid live streaming system. Simulation experiment and results are shown and discussed in Sec.IV. We make concluding and prospect remarks in Sec.V.

II. CDN–P2P-HYBRID OVERLAY TECHNOLOGY RELATED WORK ANALYSIS

1521-9097/11 $26.00 © 2011 IEEE
DOI 10.1109/ICPADS.2011.113

Using a CDN-P2P hybrid approach, client users can fetch content from the CDN edge network, from the P2P network, or from both networks. The decision of whether to use the CDN edge network or peer node resources for delivery is based on overlay network technology. Most overlay network technologies of CDN use DNS redirection technology, such as Akamai, ChinaCache; and a small number of CDN use Anycast technology, such as LimeLight. The overlay network technology of most P2P streaming application uses Tracker-based redirection technology, such as PPLive, PPStream. And many P2P file sharing applications use DHT technology, such as BT, E-mule, et al. Hybrid delivery schemes need to consider integrating these two technologies. Here we compare and analyze current main CDN-P2P-hybrid overlay network technologies as follows:

In [21], Akamai hybrid delivery scheme's overlay network mechanism mainly relies on the CDN DNS redirection mechanism. Figure1 describes the main mechanisms as follows. Every step number corresponds to the numbers in Figure1. Note Step (3) and Step (3') are alternative step; if Step (3) is selected, and then the next step is Step (4), otherwise it is Step (4').

(1) One peer node's request is directed to the CDN, which returns to the peer node a metafile. This metafile includes one or more CDN or hybrid CDN-P2P domains.

(2)Assume the metafile includes a set of domains such as peer.ake.net, peer.cdn.net, etc., each of which is resolved by the CDN DNS query mechanisms, which is authoritative for all domains returned in the metafiles.

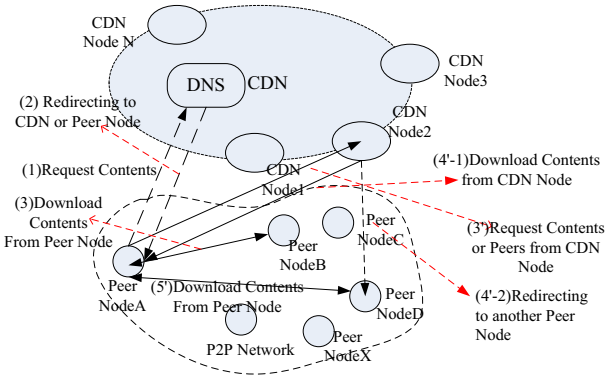

Figure1. Akamai CDN-P2P Overlay hybrid Architecture

(3) In this example, the first domain is designed to be resolved to another peer in the P2P network, and the second domain is designed to be resolved to a CDN edge server (thus acting as failover in this example). In either case, the peer node client then makes a DNS query to the first domain or sub-domain in the list, and that DNS query is resolved through the CDN DNS query mechanism to identify a nearby peer in the P2P network from which the content can be fetched.

(3') If this operation fails, if the peer cannot contact the indentified peer, or if the identified peer does not have the content, the second domain is tried, this time returning an nearby edge server in the CDN. The edge server can then choose how to handle this request, ie., (4'-1) by delivering the content objects itself, or (4'-2) redirecting the request to a peer network resource. (5') The next step of (4'-2) would be to download contents from a peer node.

In Livesky[16], hybrid delivery scheme's overlay route mechanism firstly relies on the CDN DNS redirection mechanism to reach a edge server, and the edge server can act as the tracker of P2P overlay to help the requested peer find neighbors. They described the main mechanisms as the following Figure2.

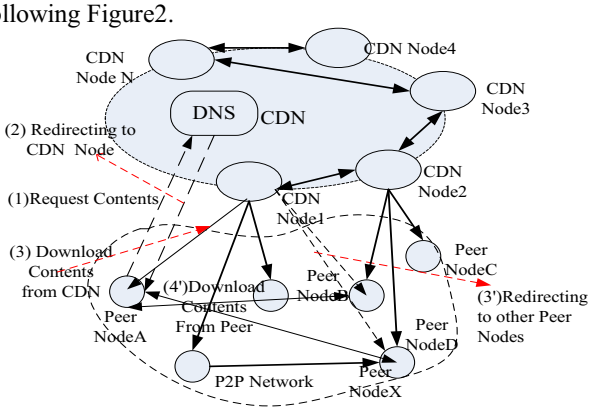

Figure2. LiveSky CDN-P2P Overlay hybrid Architecture

(1) A client first obtains the URL for the live stream from the content source (e.g., livesky://domainname/live1).

(2) The GSLB component of the CDN takes into account the client location, the edge SN (Service Node-CDN Node in Figure2) location, and the edge SN loads, to find a suitable edge SN for this client. The client is then redirected to the edge SN using traditional DNS-based redirection techniques.

(3)The edge SN acts as a seed for the P2P operation for the LiveSky-enabled clients assigned to it.

(3') The edge SN serves as a tracker for the P2P operation to bootstrap new clients with candidate peers.

(4') And then the peer node will request and download content from other peer nodes.

In [20], Hung-Chang Yang et al. proposed a two step selection approach on landmark-based selection algorithm to find the nearest replica-cache server (CDN Node) and peer-caches to download content in this architecture. We call it Two-step-landmark. Figure3 describes the main mechanisms of two-step-landmark.

(1) The first step also depends on DNS to find the nearest replica-cache server. When the end user wants to get the content distribution service site via querying the local DNS server, the content provider server issues the measure messages to the CDN-DNS, which act as landmarks. The local DNS server starts up the querying process to the CDN-DNS, the CDN-DNS will measure the round-trip time between the CDN-DNS and the local DNS server. Note: Figure3 merges the interaction between local DNS and CDN-DNS.

(2) When the local DNS server receives responses from all CDN-DNS, it replies to the end user which is the nearest replica-cache servers. If there are many replica-cache servers belonging to the same region, they can further use Euclidean-Distance equation to compute which is the nearest replica-cache server.

(3) At the second step, when the end users contact the nearest replica-cache server, the replica-cache server issues

the measure messages to other peer-caches which act as landmarks. The landmark will measure the round-trip time between the landmark and the end user.

(4) When the replica-cache server receives responses from all landmarks, it replies to the end user which is the nearest peer-cache.

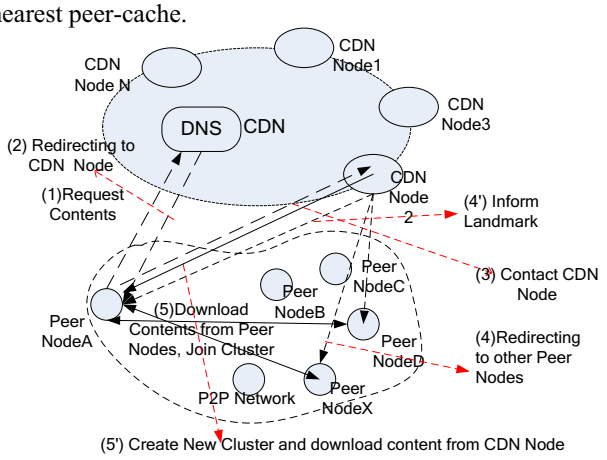

Figure3. Two-step-landmark CDN-P2P Overlay hybrid Architecture

(5) The end user will send joining messages to the nearest peer-cache and the nearest peer-cache will add the end user to the cluster member list. The end user downloads contents from peer-cache.

(4') Otherwise, if the end user is out scope of the nearest peer-cache, the end user will create a cluster for itself and inform the nearest replica-cache server.

(5')Further, the end user will act as a cluster leader to inspire new peers, and download contents from replica-cache.

In [13], Jie Wu et al. proposed PeerCDN architecture. Just as Figure4 described, PeerCDN is a two-layer streaming architecture. The upper layer is a CDN framework layer. The lower layer is called a Peer node layer, which consists of multiple groups of clients who request the streaming services. The connecting point between the CDN framework layer and the Peer node layer is located in the nearest replica server: the 'Strong Node'. It is the Strong Node's duty to coordinate the content direction for each request.

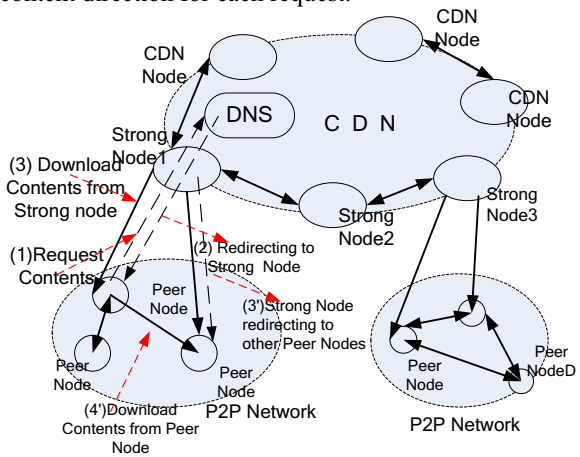

Figure4. PeerCDN CDN-P2P Overlay hybrid Architecture

Figure 4 illustrates the content request flow; strong node 2 is a candidate of strong node 1 and strong node 3.

(1)Request Contents from DNS (here merging the interaction between local DNS and CDN-DNS).

(2) DNS Redirecting to Strong Node.

(3) Download Contents from Strong node.

(3')Optionally, Strong Node redirecting to other Peer Nodes.

(4')Download Contents from selected Peer Nodes.

III. A Novel Live Streaming Service Scheme through Coordinating CDN and P2P

Here, based on the above analysis, we design a novel live streaming service scheme through coordinating CDN and P2P.

A. CDN and P2P Overlay Hybrid Architecture Design

In our above analysis of the four methods, Akamai, LiveSky, Two-step-landmark, PeerCDN, their common feature is a CDN acting as a starting point, the users firstly connect to the CDN DNS server, and then are redirected to the CDN nodes/strong nodes These CDN nodes have the P2P-Tracker-like function, they can further direct users to other peer users. Through such methods, P2P system is attached to the CDN system. These hybrid methods are efficient for building P2P-aided CDN systems, but they disrupt the native overlay organizational mechanisms of P2P. For example, PPLive and PPStream are two largest P2P streaming media operators, they use their own tracker, not using CDN's DNS to guide peer users. If PPLive and PPStream want to integrate with CDN through such method, PPLive and PPStream have to break their existing overlay construction and data transmission mechanism to adapt to CDN, which will increase the integration difficulty.

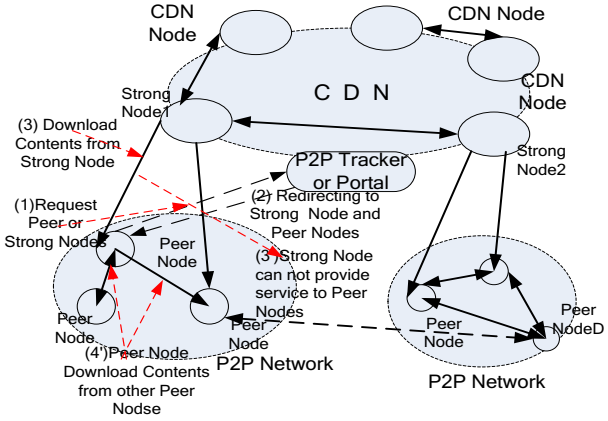

Figure5. Our Novel CDN and P2P Overlay hybrid Architecture

Different from the above methods, we use a smoother CDN and P2P overlay hybrid approach. Just as the Figure 5 described, the overlay network of our novel CDN-P2P-hybrid architecture has two layers: the first layer is the CDN network layer from the center to edge of the network, and the second layer is the P2P overlay network layer. We do not use the CDN's DNS redirection mechanism, but we let some CDN nodes acting as Strong Nodes firstly join P2P overlay network to become most stable nodes to provide content service. When peer nodes enter the overlay network, they

can be redirected to some Strong Nodes and some common peer nodes through a P2P tracker or portal. When a Strong Node receives incoming peer nodes' requests, it makes decisions through some scheduling algorithms. If a peer node's request meets the service conditions of the Strong Node, for example, it is one of the earliest-arriving peer nodes or one starved peer node in an emergency, the Strong Node may directly provide content to it. If the Strong Node can not provide services, this peer node can request content from other peer nodes. When some peer nodes are directed the same strong node, they naturally belong to the same region where this Strong Node located. Through this method, the peer nodes gradually belong to different regions, while at the same time everyone can contact other peer nodes of other regions if needed.

Figure 5 illustrates the peer node content request process:

(1) Peer requests Contents from P2P portal or tracker.

(2) P2P portal or tracker redirects it to a CDN Strong Node and some peer nodes.

(3) The CDN Strong Node will use some filtering algorithm to select some node to serve. If one peer node gets permission, the peer node downloads contents from the CDN Node.

(3') Otherwise, CDN Strong Node can't provide service to this peer and this peer is refused.

(4')The peer node downloads contents from other peer nodes.

B. Peer Node Hybrid-Buffer Architecture Design

In the current P2P live streaming system, the peer client playing buffer is a mixed zone for obtaining data block and contributing data block at the same time. The buffer at each peer represents a sliding window of the media channel, containing blocks to be played and shared in the immediate future. P2P has two kinds of typical overlay structures: tree-based overlay and mesh-based overlay. These two methods have different buffer filling methods.

In the tree-based approach, an overlay construction mechanism organizes participating peers into a single tree or multiple trees. Each peer determines a proper number of trees to join based on its access link bandwidth [24]. The content delivery is a simple push mechanism from the tree's upper layer node to the tree's lower layer node's playing buffer.

In the mesh-based approach, participating peers form a randomly connected overlay - a mesh. Each peer tries to maintain a certain number of parents (i.e., incoming degree) and also serves a specific number of child peers (i.e., outgoing degree) [24]. Upon arrival, a peer contacts a tracker server to receive a set of peers that can potentially serve as parents, and then the peer will pull contents from some selected peers into its playing buffer. Usually, they use greedy algorithm to pull data and fill the playing buffer.

From the users experience's point of view, a good QoS for P2P live streaming requires a fast startup time and less freezing. A fast startup time requires the user to receive a few continuous blocks at the beginnings of the media content in around 15-20 seconds. Therefore, how to reasonably and quickly fill the buffer is a key factor. Currently, in most of CDN-P2P-Hybrid delivery architecture, peer nodes greedily request CDN to fill their buffer when they need to startup or encounter an emergency situation. But this method is easy to cause overload of CDN, especially flash crowd occurs. Our designed peer node buffer architecture considers the appropriate and rational use of CDN resource.

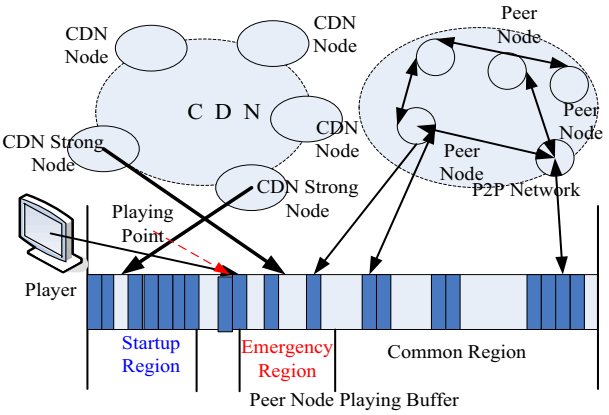

Figure6. Peer Node Buffer Architecture

As the figure6 shows, the buffer is divided into three regions: Startup Region, Emergency Region, and Common Region, which is the video playback order from left to right. The descending priority of three regions is: Startup Region →Emergency Region → Common Region.

Startup Region: the startup area. P2P nodes can obtain continuous blocks from the CDN Strong Node to startup fast playing. Its length is 1 / 8 of the total buffer length.

Emergency Region: 10 seconds data after the current playing point. If there are pieces missing in this part, the node also can directly send block request to the CDN Strong node; at the same time, it can also send the block request to its superior neighbors.

Common Region: This part has the lowest priority, referring to the all other part except the Startup, Emergency Region. The peer will request the common region pieces from its neighbors. If the peer cannot get all common region blocks in time, some blocks will become Emergency Region blocks as the playing point forward.

In the CDN-P2P hybrid scenario, an important issue to consider is the size of data blocks. In general, the data block size of CDN and P2P delivery is different. For example, UUSee Inc. is one of the leading P2P multimedia solution providers in mainland China, it simultaneously broadcasts over 800 live streaming channels to millions of peers, mostly encoded to high quality streams around 500 Kbps. The peer buffer size in UUSee is 500 media blocks, and each block represents 1/3 second of media playback [5]. So we can calculate that the size of each block is approximately 20.8KB. But in each transmission, CDN is capable of providing larger size blocks, such as 1MB- 16 seconds data for each request. If so, we need to consider CDN and P2P distributing different sizes of blocks to fill the buffer. Now most of the hybrid schemes do not yet consider this issue.

C. Identifying and Serving Super Node Algorithm

But we need to know, CDN Strong nodes' service capabilities are also limited. If all the peer nodes request data

from Strong Nodes in the start-up and emergency situations, the Strong Nodes will become overloaded. Therefore, Strong Nodes need some filtering mechanism to select partial nodes from all visiting nodes to provide services. Zimu Liu et al.[5] proposed Distilling Superior Peers in Large-Scale P2P Streaming Systems. In large-scale P2P live streaming systems with a limited supply of server bandwidth, increasing the amount of upload bandwidth supplied by peers becomes critically important. Intuitively, two types of peers are preferred to be kept up in a live session: peers that contribute a higher percentage of their upload capacities, and peers that are stable for a long period of time. The fundamental challenge is to identify these types of "superior" peers in a live streaming session [5].

Feng Wang et al.[6] found that the significantly longer lifespans of the stable nodes allow each of them to appear in much more snapshots of the overlay than transient nodes and, as a result, they constitute a significant portion (on average >70%) in every snapshot of the overlay. They introduce a Stability Index (s-Index) to characterize a node's degree of stability: an s-Index close to 0 means the node is highly transient and close to 1 means it is likely stable.

We identify super nodes in a light-weighted way. Here, we use the CDN Strong node collecting partial P2P real-time status data in its region, including average download rate, average upload rate, out-degree, already-played blocks, to determine whether one peer node is a super node. When these super nodes get content first, they can better and faster serve the other peer nodes. We think that the capability of a peer node in the P2P streaming system depends on its static capability and dynamic capability. Static capabilities include its upload bandwidth capacity and download bandwidth capacity. Dynamic capabilities include playback quality, node duration time in the session so far, the average incoming rate and outgoing rate. We evaluate the overall capability of a node in Δt period as the following formula (1)-(4):

$$ODC_i = \frac{\sum_{\Delta t} Degree^i_{out}}{Max(\sum_{\Delta t} Degree^j_{out}, j \mid j \in PList_{CDN})} \quad (1)$$

$$SDC_i = \frac{\sum_{\Delta t} Data^i_{out} / (\Delta t)}{Max(\sum_{\Delta t} Data^j_{out} / (\Delta t), j \mid j \in PList_{CDN})} \quad (2)$$

$$PQ_i = \frac{\sum_{\Delta t} Blk^i_{played}}{\sum_{\Delta t} Blk^i_{should}} \quad (3)$$

$$PUC_i = \alpha \times ODC_i + \beta \times SDC_i + \gamma \times PQ_i \quad (\alpha + \beta + \gamma = 1) \quad (4)$$

$\sum_{\Delta t} Degree^i_{out}$ is a node's out degree in sampling Δ t period; $Max(\sum_{\Delta t} Degree^j_{out}, j \mid j \in PList_{CDN})$ is the maximum out degree of one CDN region's all nodes in sampling Δt period; The ratio between them is ODCi, the value of a node's out degree capacity in sampling Δ t period. $\sum_{\Delta t} Data^i_{out} / (\Delta t)$ is the average data sending rate in Δ t period; $Max(\sum_{\Delta t} Data^j_{out} / (\Delta t), j \mid j \in PList_{CDN})$ is maximum data sending rate of one CDN region's all nodes in Δt period; the ratio between them is SDCi , the node's sending data capacity in sampling period. The $\sum_{\Delta t} Blk^i_{played}$ means how many blocks the node has played in sampling Δt period and $\sum_{\Delta t} Blk^i_{should}$ denotes how many blocks the node should play in sampling Δt period. The ratio between them can tell us one node's playback quality PQi in sampling Δt period. All nodes belong to PList $_{CDN}$, which is a peer collection in a CDN management region, including direct contacting peers and indirect reporting peers. It is obvious that the node dynamic overall capability PUCi is ODCi, SDCi and PQi 's weighted average value. α , β , andγ can be adjusted according to different scenarios. It is noted that when a new node joins in the session, CDN judges its capability only from its upload capacity because CDN doesn't have the statistic of its dynamic overall capability.

When one CDN Strong Node finds that there are too many concurrent peer requests coming during a short time, beyond its service capacity, such as flash crowd, it uses the above filtering algorithm to select some super nodes to serve. However, if we only consider serving super node, it means that CDN node can never serve common peer node. Therefore, after we calculate peer node capacity, we need to take into account the current load of CDN node. We use the following formula (5) to make a tradeoff between the peer node capacity and the CDN node current load.

$$PUC_i \geq \delta \frac{R^{CDN}_{out}}{BW^{CDN}_{out}} \quad (5)$$

When the above inequation is true, the CDN server will serve the node. The CDN node's current load rate is the ratio of outgoing rate: R^{CDN}_{out} to its outbound (upload) bandwidth: BW^{CDN}_{out}. δ is a dynamic coefficient, it will change according to the ratio. If the ratio decreases, we consider that the value of δ is too large, so we can decrease its value, and vice versa. We can see that the higher the peer node's capacity is, the greater its possibility of getting CDN services is. But when a CDN node' pressure is not heavy, ordinary nodes can also get services from CDN. For example, the right of inequation is 0.3, illustrating CDN node' load is light; when a peer node's capability is 0.35, not very superior, but it also can get CDN service, as 0.35 > 0.3 (the inequation is true).

The following pseudo-codes describe our key algorithms of CDN Strong Node selecting super or common peer nodes to serve for startup or emergency situation based on its load.

TABLE I. ALGORITHM 1: SERVING-STARTUP()

1: if $PUC_i \geq \delta(R^{CDN}_{out} \div BW^{CDN}_{out})$ then

```
2:  req_seq <-- min(n_startup_pack_num ,
cdn_node.m_head_recved_seq_num)
3:  for seq_num in req_seq
4:   if seq_num not in buffermap then
5:    req_block.push(seq_num)
6:   end if
7:  end for
8:  send(cdn_node, req_block)
9: end if
10: tracker.startup_num ++
```

TABLE II. ALGORITHM 2: SERVING-EMERGENCY()

```
1: if m_neighbors.size() < MIN_NEIGHBOR then
2:  if not m_neighbors.find(cdn_node) and
PUC_i ≥ δ(R_out^CDN ÷ BW_out^CDN)  then
3:   m_neighbors.add(cdn_node)
4:   /*Add cdn node into neighbor list, and make a require packet
scheduler again*/
5:   req_block_scheduler();
6:   for neighbor in m_neighbors
7:    send(neighbor.id,neigbor.req_block)
8:   end for
9:  end if
10: end if
11: tracker.emergency_num ++
```

TABLE III. ALGORITHM 3: REQ_BLOCK_SCHEDULER()

```
1: req_block_scheduler()
2: for seq_num in head_known_seq_nums then
3:  if seq_num not in buffmap then
4:  /* If seq_num is close to the play-back point, then it is an emergent
block. */
5:  if seq_num - play_back_num < MIN_EMERGENCY_NUM and
PUC_i ≥ δ(R_out^CDN ÷ BW_out^CDN)  then
6:   master_cdn.req_block.push(seq_num)
7:  end if
8: else
9: /* Get list of neigbors who owns the block. */
10: neighbors <-- get_holders(seq_num)
11: neighbor <-- random(neighbors)
12: neighbor.req_block.push(seq_num)
13: end else
14: end if
15: end for
```

IV. SIMULATION EXPERIMENTS AND ANALYSIS

A. Simulation Experiment

To evaluate the performance of our system, we modify a P2P streaming simulator [25] developed by Meng Zhang. The simulator itself supports a P2P-ONLY pull-based live streaming scheme. We add the CDN layer into this simulator. We compare our CDN-P2P-hybrid pull-based live streaming scheme with a P2P-ONLY pull-based live streaming scheme.

The revised simulator can configure multiple CDN nodes as strong nodes to simulate a CDN-P2P-hybrid system. A strong node will have much greater upload bandwidth (10Mbps), and maintain a larger neighbor list to serve peer nodes (When the playing rate is 300kbps, one CDN strong node can serve 30 peer nodes concurrently). All the CDN strong nodes join the system immediately when the live streaming system begins to run, and they can provide all blocks which peer nodes require. Each arriving peer node will be redirected to the CDN strong node which has the minimum delay to that peer node. Therefore, all peer nodes will be divided into different region. Each region is lead by a CDN strong node.

Just as Part III describes, one peer node's buffer most important two zones are startup region and emergency region. CDN strong nodes will divide their upload bandwidth into two parts. One is for peer nodes' startup region data, the other is for peer nodes' emergency region data. This ensures that the strong node can simultaneously serve the startup data and emergency data. These two kinds of data service shares will be decided by the tracker. The tracker will get the number of starting nodes and emergent nodes in every 5 second intervals. Then, it will calculate the current startup and emergency coefficient according to this formula (6):

$$CO^{t}_{startup} = \mu CO^{t-1}_{startup} + N_{startuup}$$

$$CO^{t}_{emergency} = \mu CO^{t-1}_{emergency} + N_{emergency}$$

$$\mathrm{P}_{startup} = \frac{CO^{t}_{startup}}{CO^{t}_{startup} + CO^{t}_{emergency}} \qquad (6)$$

$$\mathrm{P}_{emergency} = \frac{CO^{t}_{emergency}}{CO^{t}_{startup} + CO^{t}_{emergency}}$$

Last- time-startup-coefficient $CO^{t-1}_{startup}$ is added to the formula to represent the influence of historical data, and μ is the influence factor. $N_{startup}$ is starting node number. The emergency coefficient is calculated similarly. These two coefficients are then used to divide the proportion of the strong node's upload bandwidth for startup data and emergency data: $P_{startup}$ and $P_{emergency}$ When the CDN strong node's load is light, the CDN node will serve startup data and emergency data to all requesting common peer nodes. While the CDN strong node's load becomes heavy, the CDN node will use the above filtering algorithm to transfer the startup data and emergency data to those selected super nodes.

The P2P-ONLY live streaming scheme uses purely random neighbor selection and random packet assignment among neighbors. Each peer node randomly selects a certain number of nodes as its neighbors from the tracker. The peer node will scan its buffer to find the absent blocks at regular intervals. If it needs blocks, it will select one neighbor node which has these blocks, and then send a request to this node. In our experiment, we use the following three typical metrics to evaluate performance, which together reflect the quality of service experienced by end users.

Startup latency, which is the time taken by an end user between its requesting to join the session and receiving enough data blocks to start playback;

Playback delay, which is the time taken for a data block from being sent by the source to being played at the end user.

Ratio of supply over demand, represents the average supplying block rate for each node over the raw streaming playing rate, such as 300kbps. The former doesn't contain control overhead, which is the messages used to construct and maintain the overlay as well as the messages to exchange data availability, such as buffer-map et al.

In our simulation, the session length is set to 300 seconds; the peer node buffer time is 35 seconds; each node has 15 neighbors. The average block rate is 30 blocks per second and each block in the live streaming has the same size of

1280 bytes. We use three types of nodes, whose maximum upload bandwidth are 1Mbps, 384kbps and 128kbps respectively and the download capacities are 3Mbps, 1.5 Mbps and 768kbps. We set the fractions of the three types to 0.15, 0.39 and 0.46, respectively. We have two sets of experiments: there are 300 peer nodes joining plus 150 peer nodes suddenly appearing in the first experiment, there are 500 nodes plus 250 suddenly-joining peer node in the second experiment. In our CDN-P2P-hybrid system, we use 2 CDN nodes. Their downloading bandwidth is 2Mbps, and their uploading bandwidth is 10Mbps.

B. Experiment Result Analysis

Figure 7 shows the 300 nodes' average playback delay of the P2P-ONLY system and our CDN-P2P system during 300 seconds of session time. It is clear that, with the support of CDN-P2P-Hyrid architecture, the playback delay can be obviously reduced. In the flash crowd experiment, after 300 nodes have run for 150 second in the system, we let 150 nodes join the session between 160s and 170s. We can observe that both systems' playback delay will increase, but the CDN-P2P system has a faster reduction. The CDN-P2P system's reliability and scalability is better than P2P-ONLY.

Figure 8 shows the CDF of the startup latency of the two systems. As the peer nodes, specifically the selected super nodes, can pull more data from the CDN node at startup. Therefore, in the CDN-P2P system, 80% of the peer nodes startup in 17.5 seconds, while in the P2P-ONLY system, 80% of the peer nodes startup in 20 seconds.

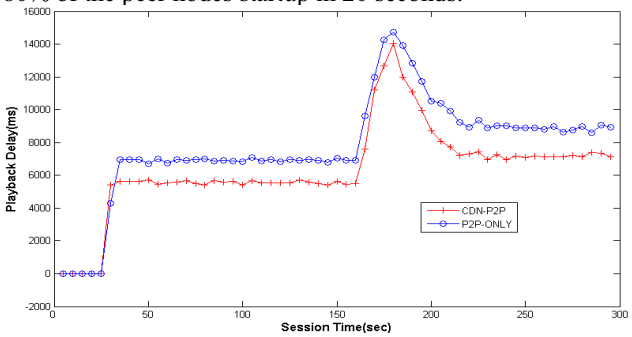

Figure7. 300 Node Playback Delay Comparison Result

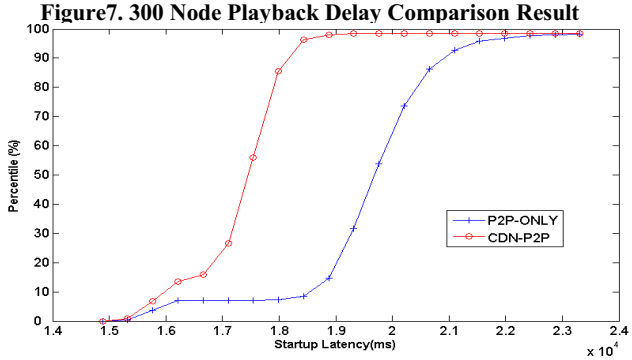

Figure8. 300 Node Startup latency Comparison Result

Figure9 shows the Ratio of supply over demand comparison results in a normal scenario and a flash crowd scenario. In the normal scenario, the ratio of the two systems is almost equal. But when 150 nodes suddenly join the session between 160s and 170s, CDN-P2P system has a higher rate of supply over demand, which illustrates our CDN-P2P system has better reliability and scalability.

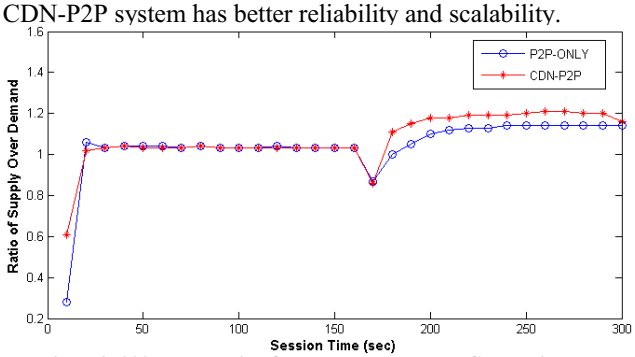

Figure9. 300 Node Ratio of supply over demand Comparison Result

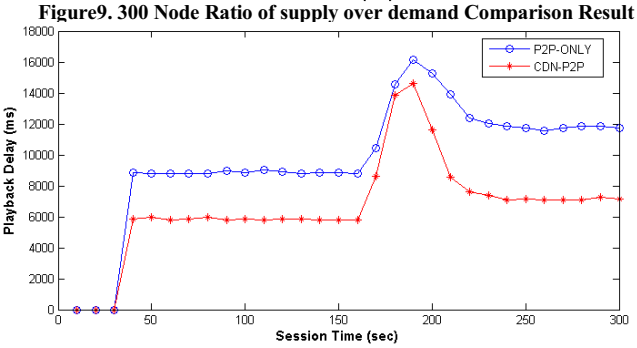

Figure10. 500 Node Playback Delay Comparison Result

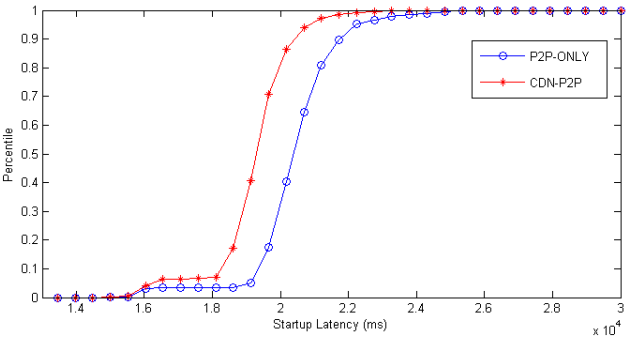

Figure11. 500 Node Startup latency Comparison Result

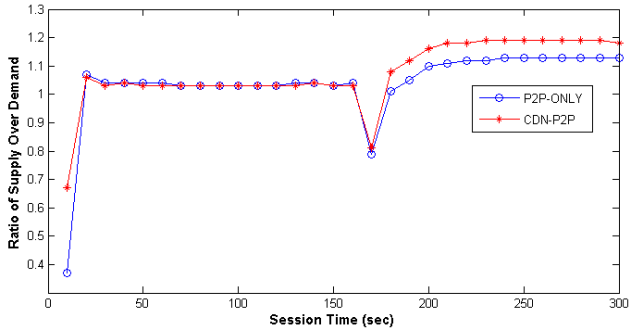

Figure12. 500 Node Ratio of supply over demand Comparison Result

Figure 10-12 show all the metrics: playback delay, startup latency and ratio of supply over demand in 500 peer nodes simulation experiments. After 500 nodes have run for 150 second in the system, we let 250 nodes join the session between 160s and 170s as a flash crowd scenario. Similar to 300 nodes case, for all three metrics, the CDN-P2P-hybrid scheme has substantially improved the live streaming

performance than P2P-ONLY system both in the reliability and scalability aspects

V. CONCLUSION AND FUTURE WORK

CDN and P2P are two main streaming delivery technologies in Internet video market, but constrained by their service models, both of them have their own strengths and weaknesses. The two techniques are complementary, so their service composition is very necessary and helpful to build a new-generation reliable, scalable, QoS-guaranteed, and large-scale streaming service platform. However, this combination is not easy to finish. A common content service platform is not a simple merging of CDN and P2P; combining CDN with P2P needs to solve some key issues. In particular, we need study the CDN and P2P integration features, specifically reliability, scalability, and controllability and so on. At the same time, Cloud computing era calls for a CDN-P2P loosely-coupled integration, which requires CDN as a service available on-demand and be utilized in an open and loosely-coupled fashion. We have begun to research a web services based CDN, P2P and VoD loosely-coupled hybrid model in [22] [29].

In this paper, we design and implement a novel reliable and scalable live streaming service system through coordinating CDN and P2P. We have conducted simulation experiments to verify our scheme superior to the general P2P streaming system. In our future work, we will further develop real prototype system to experimentally verify the advanced features of this novel scheme. Finally, we plan to carry out larger scale experiments with our prototype and make real deployment in our ongoing CNGI (China Next Generation Internet) project- National Higher Education Conference Video Resources Sharing Project, which now have contained more than 50TB academic video conference and lecture contents accessed by hundred of thousand of user.

REFERENCES

[1] Ao-Jan Su, David R. Choffnes, Aleksandar Kuzmanovic, and Fabián E. Bustamante, "Drafting Behind Akamai (Travelocity-Based Detouring)", SIGCOMM'06: 435-446.

[2] C. Huang, A. Wang, J. Li, K.W. Ross, "Measuring and Evaluating Large-Scale CDNs", Internet Measurement Conference (IMC) 2008, Greece.

[3] Yan Huang, Tom Z. J. Fu, Dah-Ming Chiu, "Challenges, Design and Analysis of a Large-scale P2P-VoD System", SIGCOMM'08, August 17–22, 2008, Washington, USA.

[4] Jiajun Wang, Cheng Huang, Jin Li, "On ISP-friendly rate allocation for peer-assisted VoD", ACM Multimedia 2008: 279-288.

[5] Zimu Liu, Chuan Wu,Baochun Li, Shuqiao Zhao, "Distilling Superior Peers in Large-Scale P2P Streaming Systems", IEEE INFOCOM 2009:82-90.

[6] Feng Wang, J. Liu, and Y. Xiong, "Stable Peers: Existence, Importance, and Application in Peer-to-Peer Live Video Streaming", IEEE INFOCOM'08.

[7] T. Karagiannis, P. Rodriguez, and D. Papagiannaki, "Should Internet Service Providers Fear Peer-Assisted Content Distribution?", in ACM/USENIX Internet Measurement Conference, IMC'2005, October 2005.

[8] Thomas Bonald, Laurent Massouli, Fabien Mathieu, Diego Perino, Andrew Twigg , "Epidemic live streaming: Optimal performance trade-offs", In. SIGMETRICS 2008.

[9] Huang, C., Li, J., & Ross, K. W. "Can internet video-on-demand be profitable?", SIGCOMM'07, 133-144.

[10] Haiyong Xie,Y. Richard Yang,Arvind Krishnamurthy et al., "P4P: Provider Portal for Applications", SIGCOMM' 08, August 17−22, 2008, Seattle, Washington, USA.

[11] David R. Choffnes and Fabian E. Bustamante, "Taming the Torrent A Practical Approach to Reducing Cross-ISP Traffic in Peer-to-Peer Systems", SIGCOMM'08, August 17−22, 2008, USA.

[12] Dongyan Xu, Sunil Suresh Kulkarni, Catherine Rosenberg, Heung-Keung Chai, "Analysis of a CDN–P2P hybrid architecture for cost-effective streaming media distribution", Multimedia Systems (2006) 11(4): 383–399.

[13] Jie Wu, ZhiHui lu, BiSheng Liu, ShiYong Zhang, "PeerCDN: A Novel P2P Network assisted Streaming Content Delivery Network Scheme", Proceeding of IEEE CIT2008, IEEE Computer Society, 601-606, 2008.7.

[14] Zhijia Chen, Hao Yin, Chuang Lin, Xuening Liu, Yang Chen, "Towards a Trustworthy and Controllable Peer-Server-Peer Media Streaming: An Analytical Study and An Industrial Perspective", GLOBECOM 2007.

[15] Xuening Liu, Hao Yin, Chuang Lin, Yu Liu, Zhijia Chen, Xin Xiao, "Performance Analysis and Industrial Practice of Peer-Assisted Content Distribution Network for Large-Scale Live Video Streaming", AINA 2008: 568-574.

[16] Hao Yin, Xuening Liu, Tongyu Zhan, Vyas Sekar, Hui Zhang, "Design and Deployment of a Hybrid CDN-P2P System for Live Video Streaming: Experiences with LiveSky",ACM Multimedia 2009.

[17] Cheng Huang,Angela Wang, "Understanding Hybrid CDN-P2P:Why Limelight Needs its Own Red Swoosh", NOSSDAV '08 Braunschweig, Germany.

[18] Hai Jiang, Jun Li, Zhongcheng Li, et al, "Efficient Large-scale Content Distribution with Combination of CDN and P2P Networks", International Journal of Hybrid Information Technology, 4. 2009.

[19] Duyen Hoa HA,Thomas Silverton,Olivier FOURMAUX, "A novel Hybrid CDN-P2P mechanism For effective real-time media streaming", www-rp.lip6.fr/~fourmaux/Stages/HA.ACM_Rapport.pdf.

[20] Hung-Chang Yang, Min-Yi Hsieh, Hsiang-Fu Yu, Li-Ming Tseng, "A Replication-Aware CDN-P2P Architecture Based on Two-Step Server Selection and Network Coding", PCM 2008: 738-747.

[21] Michael M.Afergan, et al., "Hybrid content delivery network (CDN) and peer-to-peer (P2P) network", United States Patent Application 20080155061.26, Assignee: Akamai, www.freepatentsonline.com/y2008/0155061.html,Jun 2008.

[22] ZhiHui Lu Jie Wu Chuan Xiao WeiMing Fu YiPing Zhong, "WS-CDSP:A Novel Web Services-based Content Delivery Service Peering Scheme", Proceeding of 2009 IEEE SCC 2009, IEEE Computer Society, 348-355, 2009.9.

[23] Feng Wang, Yongqiang Xiong, Jiangchuan Liu, "mTreebone: A Hybrid Tree/Mesh Overlay for Application-Layer Live Video Multicast", IEEE ICDCS 2007.

[24] Nazanin Magharei, Reza Rejaie,Yang Guo, "Mesh or Multiple-Tree: A Comparative Study of Live P2P Streaming Approaches", IEEE INFOCOM 2007.

[25] Peer-to-peer Streaming simulator, Meng Zhang, http://media.cs.tsinghua.edu.cn/~zhangm/download/p2pstrmsim_allin one.tar.gz, 2008.

[26] Om Malik, "Grid Joins the P2P CDN Party", http://gigaom.com/2007/10/30/grid-networks/,Oct. 30, 2007.

[27] Shicong Meng, Ling Liu, Jianwei Yin, "Scalable and Reliable IPTV Service Through Collaborative Request Dispatching,"In Proceedings of ICWS. 2010, 179-186.

[28] CDNs and P2P, http://www.pandonetworks.com/node/185, 2007.

[29] ZhiHui Lu, Jie Wu, WeiMing Fu,"Towards A Novel Web Services Standard-supported CDN-P2P Loosely-coupled Hybrid and Management Model", Proceeding of 2010 IEEE International Conference on Services Computing (SCC 2010), IEEE Computer Society, 297-304, 2010.