applied sciences

MDPI

Article

An Adaptive Buffering Scheme for P2P Live and Time-Shifted Streaming

Eunsam Kim [1], Taeyoung Kim [2] and Choonhwa Lee [3,*]

1 Department of Computer Engineering, Hongik University, Seoul 04066, Korea; eskim@hongik.ac.kr
2 Pasoo Inc., 396 Worldcup bukro, Mapo-gu, Seoul 03925, Korea; kimty0@fasoo.com
3 Division of Computer Science and Engineering, Hanyang University, Seoul 04763, Korea
* Correspondence: lee@hanyang.ac.kr; Tel.: +82-2-2220-1268

Academic Editor: Arun Somani
Received: 30 November 2016; Accepted: 15 February 2017; Published: 18 February 2017

Abstract: Recently, P2P streaming techniques have been a promising solution to a large-scale live streaming system because of their high scalability and low installation cost. In P2P live streaming systems, however, it is difficult to manage peers' buffers effectively, because they can buffer only a limited amount of data around a live broadcasting time in the main memory and suffer from long playback lag due to the nature of P2P structures. In addition, the number of peers decreases rapidly as the playback position moves further from this time by performing time-shifted viewing. These situations widen the distribution of peers' playback positions, thereby decreasing the degree of data duplication among peers. Moreover, it is hard to use each peer's buffer as the caching area because the buffer area where the chunks that have already been played back are stored can be overwritten at any time by new chunks that will arrive soon. In this paper, we therefore propose a novel buffering scheme to significantly increase data duplication in buffering periods among peers in P2P live and time-shifted streaming systems. In our proposed scheme, the buffer ratio of each peer is adaptively adjusted according to its relative playback position in a group by increasing the ratio of the caching area in its buffer as its playback position moves earlier in time and increasing the ratio of the prefetching area as its playback position moves later. Through extensive simulations, we demonstrate that our proposed adaptive buffering scheme outperforms the conventional buffering technique considerably in terms of startup delay, average jitter ratio, and the ratio of necessary chunks in a buffermap.

Keywords: P2P live streaming; time-shifted viewing; adaptive buffering; prefetching; caching

1. Introduction

Advances in digital technologies and computer networks have enabled Internet Protocol television (IPTV) systems that provide multimedia services based on IP networks. As the number of IPTV subscribers increases and personalized services such as Video-on-Demand (VOD) and time-shifted TV become more common, the conventional server architectures based on Content Distribution Networks (CDNs) have been limited in scalability because of high installation costs. Therefore, highly scalable peer-to-peer (P2P)-based IPTV systems that can be implemented at low installation cost have emerged as an alternative to conventional systems [1,2]. In mesh-based structures that have been employed by most commercial P2P streaming systems, peers share data required for video playback [3–8]. To do so, they exchange buffermaps with their neighbor peers to find which peers have the necessary chunks. Each peer then receives these chunks from multiple neighbor peers simultaneously after requesting each one.

To improve the performance of P2P streaming systems, many buffering techniques (e.g., prefetching and caching) have been proposed [9–18]. Especially in P2P VOD systems, once peers store video files that they want on their own storage devices, they share these files with each other at any time. Therefore, as long as each peer finds neighbor peers that can transmit the amount of data required to playback a desired video until it completes playback, it is not difficult to perform buffering techniques effectively, because each neighbor peer stores an entire part of the video file. Moreover, even though playback speed changes when supporting video cassette recorder (VCR) operations such as fast forward and rewind, neighbor peers can also transmit a sufficient amount of data because they already have the relevant data stored and access patterns can be easily predicted.

In P2P live streaming systems, however, since peers can playback a broadcast video after receiving data generated in real time, they can buffer only a limited amount of data around the live broadcasting time in the main memory. To playback the video seamlessly, it is therefore important to increase data duplication among participating peers by making their buffered data overlap as much as possible. However, due to the nature of P2P live streaming, peers suffers from playback lag between a source server and peers. In other words, even though peers request to move to a live broadcasting position, they may playback data a significant time after the server has transmitted them, depending on when they joined. As a result, even the playback positions of the peers watching a live broadcast are dispersed. Moreover, since playback requests in a live streaming system tend to be concentrated around a live broadcasting time [19], the number of peers decreases rapidly as the playback position moves further from this time by performing time-shifted viewing through rewind, pause, and slow motion functions. Since this widens the distribution of peers' playback positions, data duplication among peers decreases further. If a peer cannot find neighbor peers that can transmit the data required to playback its desired video, peers whose buffering periods do not overlap with that of the given peer should be selected as neighbor peers. This implies that a peer must skip a certain period to synchronize the buffering periods of its neighbors with that of its own, thereby resulting in the degradation of playback quality in P2P live streaming systems.

On the other hand, since each peer's buffer behaves like a finite circular queue in conventional P2P live streaming systems (as shown in Figure 1), the buffer area where the chunks that have already been played back are stored can be overwritten at any time by new chunks arriving soon. Thus, such old chunks stored in the buffer area are usually considered as unavailable. This is because it is possible that—even though neighbors request old chunks—those chunks can become invalid immediately before the peer actually starts transmitting them. Therefore, it is difficult to use each peer's buffer as the caching area to avoid accessing invalid data.

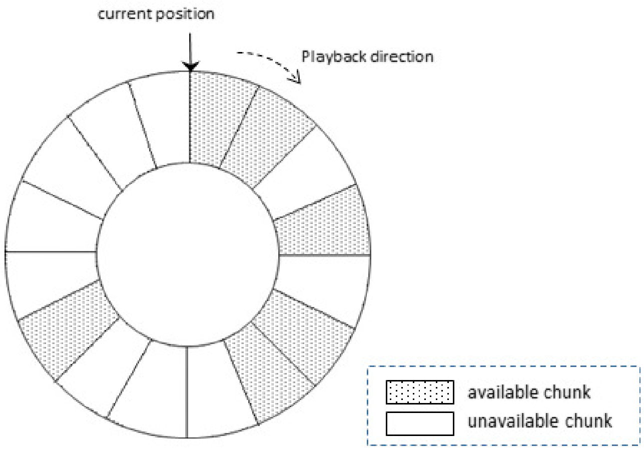

Figure 1. Buffer status of a peer in peer-to-peer (P2P) live streaming systems.

In this paper, we therefore propose a novel buffering scheme in P2P live and time-shifted streaming systems to significantly increase data duplication in buffering periods among peers by adaptively adjusting the ratio of the caching and prefetching areas in the buffer. To this end, we form groups of peers according to their playback positions and select a super peer from each peer group. A super peer periodically gathers playback position information of all peers belonging to its group. In P2P live streaming systems, each peer should receive data from those whose playback lags are shorter than its own (i.e., those whose playback positions are earlier in time since they have joined earlier). Thus, the peer whose playback position is latest among all peers belonging to a group does not need to cache data that have already been played back, because all the other peers in the group have also already played them back. That is, almost all areas of its buffer need to be used to prefetch subsequent chunks. On the contrary, the peer whose playback position is earliest tends to supply most data that have already been played back to other peers within its group, as long as its buffering period overlaps with those of the peers. That is, the earliest peer is likely to cache the largest amount of data to provide them to other peers within its group because none of the other peers have played them back yet. In our proposed scheme, the ratio of prefetching and caching areas in the buffer of each peer is therefore determined adaptively according to its relative playback position within its group. In other words, by increasing the ratio of the caching area in the buffer of a peer as its playback position moves earlier in time and increasing the ratio of the prefetching area as its playback position moves later, the overlapping buffered data among peers within the same group can be increased significantly.

Finally, through extensive simulations, we demonstrate that our proposed adaptive buffering scheme outperforms the conventional buffering technique considerably in terms of startup delay, average jitter ratio, and the ratio of necessary chunks in a buffermap.

The remainder of this paper is organized as follows: Section 2 describes work related to buffering techniques in P2P streaming systems. Section 3 describes problems of a fixed buffering technique used in conventional mesh-based P2P systems and explains our adaptive buffering scheme in detail. Section 4 presents the experimental results of our adaptive buffering scheme in comparison with those of the conventional fixed buffering scheme. Finally, Section 5 presents our conclusions.

2. Related Work

So far, traditional client–server-based architectures have provided good performance and high availability. However, they usually require high deployment and maintenance costs. Therefore, various P2P technologies that can provide fully-distributed and cooperative applications at low cost have emerged as an alternative, thereby reducing the burdens placed on the servers [20,21]. A novel interactive broadcasting infrastructure based on a decentralized architecture has been proposed [22]. Based on a prototype digital video broadcasting-terrestrial (DVB-T) regenerative platform, this exploits P2P technology to optimize resource utilization during user-generated services' provision. A P2P Home-Box overlay structure that has enhanced home-gateways has also been proposed [23]. This structure provides means for an efficient content sharing and exchange among users based in P2P overlay. In addition, with recent advances in wireless networks and computing power, various mobile devices have been developed. Due to the heterogeneous capabilities of those mobile devices, it is essential to adapt multimedia content to enable processing with their available resources [24]. In particular, a peer-to-peer architecture for content adaptation using the MPEG-21 framework has been proposed [25]. This uses MPEG-21 digital item (DI) and digital item ddaptation (DIA) for media adaptation, and is implemented on a P2P network with super peers.

On the other hand, P2P streaming systems have been broadly classified into tree and mesh structures. In a tree-based structure, a peer receives video data from its parent peer in a push manner [26–28]. Once the tree structure is constructed, peers can transmit data at a fast rate, because they can keep transmitting data without any specific requests from their child peers. However, this structure incurs a considerable amount of overhead to rebuild tree structures whenever peers join or leave.

To address this problem, many mesh structures have been proposed [4–8]. In the mesh structures, even though a peer's neighbor leaves the network, it can still receive data from its remaining neighbors, since it receives data from several neighbor peers simultaneously. Thus, the mesh structures can provide a robust structure against peers' churn. However, since data are transmitted in a pull manner (i.e., peers receive data after explicitly requesting each chunk), transmission delays are long. As a result, the mesh structure suffers from long startup delay and playback lag. In addition, note that almost all areas of each peer's buffer are used for prefetching data that will be played back later, as mentioned above.

On the other hand, many buffering schemes have been proposed to improve the system performance in P2P VOD systems. In P2VOD, a cooperative caching scheme using a mesh-based buffermap structure has been proposed to reduce delay when a peer joins or leaves the VOD multicast tree [9]. In order to support VCR functionality efficiently in P2P VOD systems, BulletMedia reduces the load on the media server by creating replicas of the video and distributing them among several peers [10]. Unlike existing FIFO cooperative caching that determine the data to cache according to the order of requests, a video popularity-based caching scheme has also been proposed [16]. Note that, since peers store the entire part of each video on their storage devices in P2P VOD systems, the data required to playback a desired video are always available. Thus, the buffering schemes in P2P VOD systems have focused on predicting access patterns accurately and prefetching the necessary data efficiently. Unlike in P2P VOD systems, however, only a limited amount of data broadcast in real time can be buffered in the main memory in P2P live streaming systems. Therefore, existing buffering schemes for P2P VOD systems cannot be applied directly to P2P live streaming systems.

A number of buffering schemes have been thus proposed for P2P live streaming systems. However, most studies have focused on the development of an efficient overlay structure to reduce startup delay or increase playback continuity. In LayerP2P, startup delay is reduced by using a hierarchical structure of peers and a tit-for-tat algorithm [29]. A novel scheme to increase data redundancy across the system by synchronizing peers' playback positions with a media server has also been proposed [30]. However, few studies to support the time-shifted viewing in a P2P live streaming structure have been conducted. Furthermore, to the best of our knowledge, no study to control the ratio of caching and prefetching areas in each peer's buffer has been proposed.

3. Adaptive Buffering Scheme

Since live broadcast data are generated in real time in P2P live streaming systems, peers can buffer only a limited amount of data in the main memory, unlike in VOD systems. In addition, due to time-shifted viewing and playback lag caused by the nature of P2P streaming, playback positions of the peers watching time-shifted scenes as well as a live broadcast are widely spread over the entire playback period of each video, thereby decreasing data duplication among peers. On the other hand, since the chunks that have already been played back in the buffer are usually considered as unavailable, it is difficult to use each peer's buffer as the caching area to avoid accessing invalid data. In other words, it is common to use each peer's buffer only for prefetching. In this section, we therefore propose a novel buffering scheme to increase the degree of data duplication among peers in P2P live and time-shifted streaming systems by adaptively adjusting the ratio of caching and prefetching areas in the buffer of each peer according to its relative playback position within a group.

3.1. Peer Group Management

As shown in Figure 2, in our proposed P2P live and time-shifted streaming system, the tracker server assigns each period to a group consisting of peers whose playback positions are within the period after dividing the entire playback time at regular intervals. We select the peer with maximum upload capacity as a super peer from each group. The super peer manages the playback position information of all peers belonging to its group by periodically gathering the information.

Each peer asks the tracker server for the address of the super peer of the group to which it will belong when it joins the system or when it needs to join another group when its playback position changes due to time-shifted viewing. The peer then connects to the super peer to make a request to select multiple peers as its neighbors. The super peer first tries to select peers belonging to its own group. However, if the number of neighbors to satisfy the minimum playback quality is not sufficient within the group, it requests super peers in adjacent groups to select some peers belonging to those groups. Nevertheless, if the peer cannot eventually obtain a sufficient number of neighbors in the entire system, it should receive data directly from a source server.

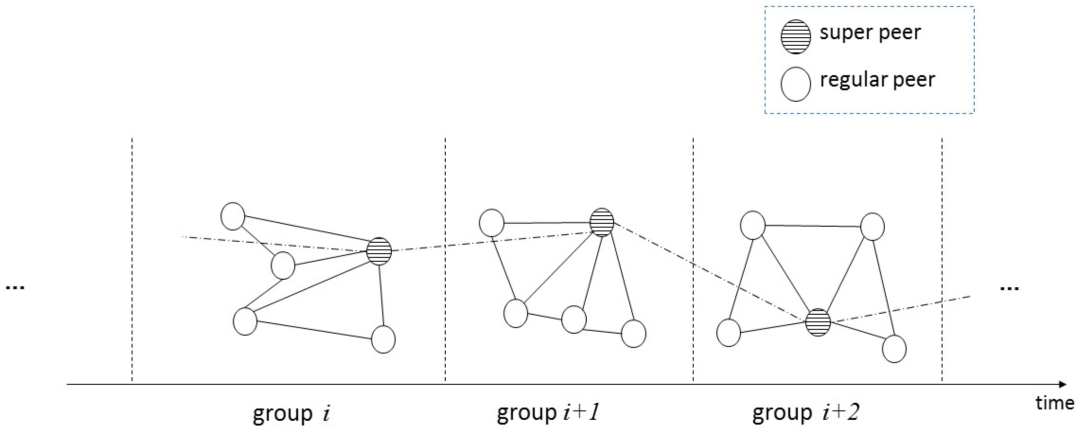

Figure 2. Peer group management according to peers' playback periods.

3.2. Determining the Ratio of Caching and Prefetching Areas

In our proposed P2P live and time-shifted streaming system, a super peer provides necessary information to peers in its group so that they can determine their ratios of the caching and prefetching areas. Figure 3 represents the buffermap information of several peers belonging to group j according to their relative playback positions. $T_{earliest,j}$ and $T_{latest,j}$ represents the current playback time positions of the earliest peer (i.e., the one closest to the live broadcasting time) and the latest peer in group j, respectively. Table 1 summarizes the symbols used in this paper.

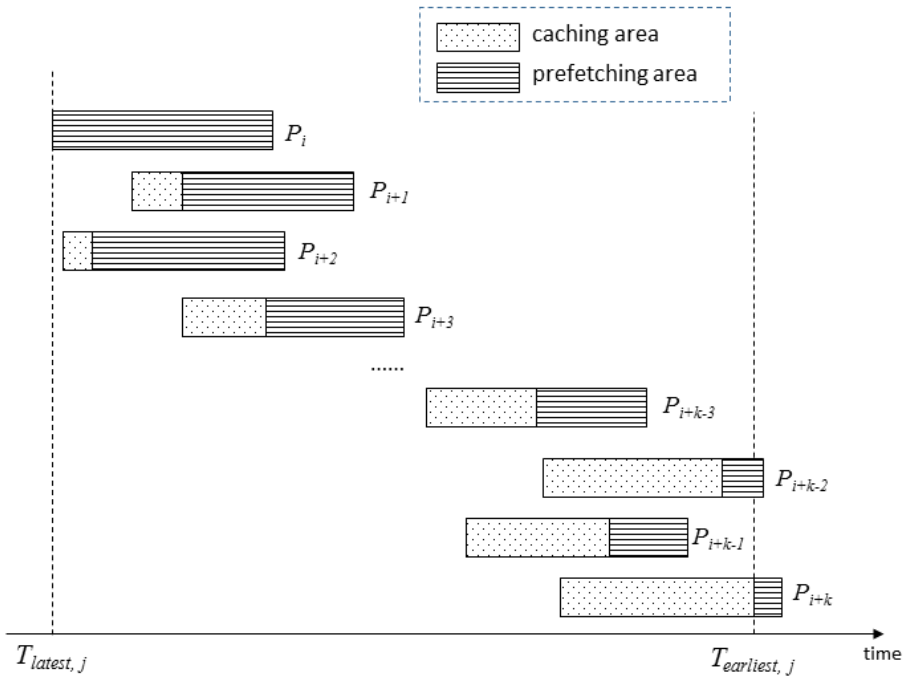

Figure 3. Ratio of caching and prefetching areas according to peers' relative playback position within group j.

Table 1. Summary of symbols.

Symbol	Definition
P_i	peer with index i
T_i	playback time position of P_i
$T_{earliest,i}$	current playback time position of the earliest peer in group i
$T_{latest,i}$	current playback time position of the latest peer in group i
α	maximum ratio of caching area
$R_{prefetch,i}$	ratio of prefetching area in P_i's buffer
$R_{cache,i}$	ratio of caching area ratio in P_i's buffer
F_i	buffer adjustment cycle of playback period i
AF_i	average cycle between playback period 1 and playback period i
UNA_i	number of peers that have joined and left per time unit between playback period 1 and playback period i
UN_i	number of peers that have joined and left per time unit during playback period i
N_i	number of peers that have joined and left during playback period i

Peers must receive necessary chunks from neighbors before a deadline for each chunk to avoid playback interruption. For example, as shown in Figure 3, P_{i+1} receives chunks from P_i, P_{i+2}, and P_{i+3} (whose buffering periods overlap with its own), and transmits its stored chunks requested by other peers. We can see that there is no peer that requires data that have been already played back by P_i (whose current playback time position is $T_{latest,j}$), since all the other peers in group j have also already played them back. Therefore, P_i does not need to cache data that have been already played back (i.e., those prior to its current playback time position) for other peers within group j. In other words, the peer whose playback time position is latest in a group does not need to cache any data.

On the contrary, P_{i+k} whose current playback position is $T_{latest,j}$ tends to transmit the largest amount of data to other peers in group j, because none of the other peers have played them back yet (i.e., because playback positions of the other peers in the group are all later than its own). This means that P_{i+k}'s buffering period is likely to overlap most with those of other peers. To improve the performance of all peers in the same group, it is therefore advantageous to use most of its buffer area excluding the minimum area for prefetching as caching area to provide data to other peers, as shown in Figure 3.

$$R_{cache,i} = \left(\frac{T_i - T_{latest,j}}{T_{earliest,j} - T_{latest,j}} \right) \times \alpha \tag{1}$$

Therefore, the ratio of caching and prefetching areas in the buffer of each peer is adaptively determined according to its relative playback position within its group, as shown in Equation (1). By increasing the ratio of the caching area as the playback position of a peer moves earlier in time and increasing the ratio of the prefetching area as its playback position moves later, we can significantly increase the overlappring buffered data among peers in a group. To this end, we first estimate how early the playback position of P_i is within group j by calculating the difference between its own playback position (i.e., T_i) and the latest playback position in group j (i.e., $T_{latest,j}$). To determine the ratio of the caching area of P_i's buffer (i.e., $R_{cache,i}$) proportional to the above result value (i.e., $T_i - T_{latest,j}$), we divide the value by the entire period calculated by subtracting $T_{latest,j}$ from $T_{earliest,j}$. In order to ensure that the earliest peer whose current playback position is $T_{earliest,j}$ can obtain the minimum prefetching area, we also multiply $((T_i - T_{latest,j})/(T_{earliest,j} - T_{latest,j}))$ by parameter α to limit the $R_{cache,i}$ up to α, as shown in Equation (1).

$$R_{prefetch,i} = 1 - R_{cache,i} \tag{2}$$

Since the prefetching area of P_i's buffer is the remaining portion excluding the caching area, $R_{prefetch,i}$ is calculated by using Equation (2).

Figure 4 shows the ratios of caching and prefetching areas when employing our proposed adaptive buffering scheme and the conventional fixed buffering scheme. Note that the two same peers in both cases have the same playback position while having different ratios of caching and prefetching areas. Since the ratio of caching and prefetching areas in a peer's buffer is almost fixed in the conventional fixed buffering scheme, it does not need to consider the buffering status of other peers in the group. On the contrary, since, in our adaptive buffering scheme, the ratio of the caching and prefetching areas in each peer's buffer can be adjusted adaptively according to its relative playback position within a group, it caches more data to provide to other peers as its playback position moves earlier in time.

For example, in Figure 4, the number of chunks that P_i can provide to other peers is two in the fixed buffering scheme: one chunk cached in the caching area of its buffer and the other in the prefetching area received from an earlier peer. However, in the adaptive buffering scheme, since the ratio of the caching area of P_i increases, five chunks in the caching area can be provided to other peers. Therefore, the adaptive buffering scheme can improve the overall performance, because it can increase the degree of data duplication among peers by increasing the overlapping periods among peers' buffers even though the intervals among peers' playback positions increase.

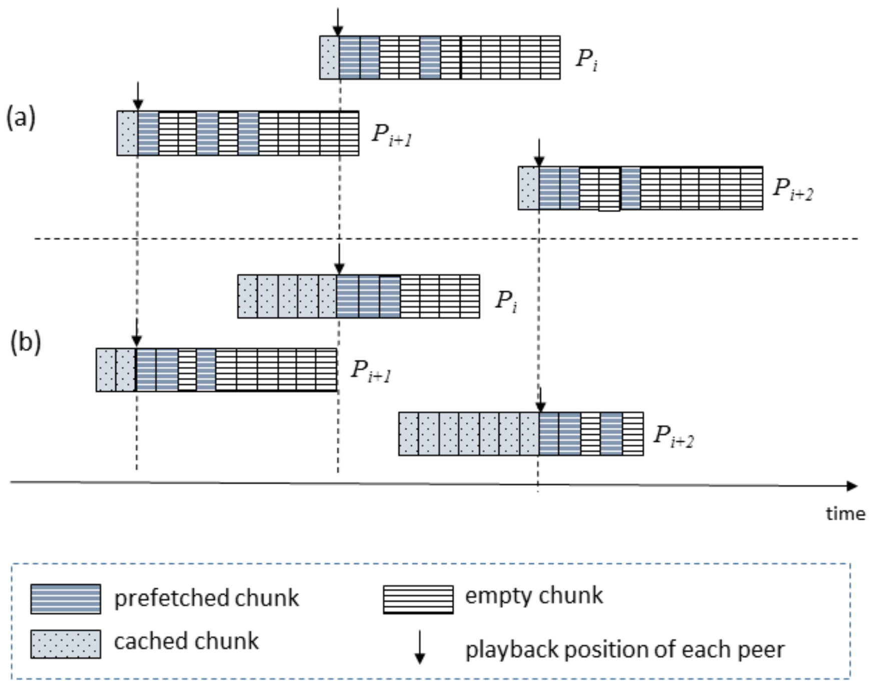

Figure 4. A comparison of ratios of caching and prefetching areas between (**a**) fixed buffering scheme and (**b**) adaptive buffering.

3.3. Adjustment Cycle of Buffer Area Ratio

Since peers frequently join and leave P2P networks and perform time-shifted viewing, membership of each group changes dynamically in the adaptive buffering scheme. When a peer joins a group, it determines the ratio of the caching and prefetching areas in its buffer after receiving the necessary information from its super peer. However, a peer that has stayed in the same group for a long time periodically needs to adjust the ratio of its buffer areas to reflect changed group membership, since some peers may have joined and left the corresponding group.

In our adaptive buffering scheme, we determine the cycle for adjusting the buffer area ratio of each peer so that, as membership in a group changes more frequently, its buffer area ratio can be adjusted more often. When the previous buffer ratio adjustment period ends, the next buffer ratio adjustment cycle is determined depending on how many peers have joined and left the group in the previous period.

$$F_i = AF_{i-1} \times \frac{UN_{i-1}}{UNA_{i-1}} \quad \left(AF_{i-1} = \frac{\sum_{k=1}^{i-1} F_k}{i-1}, \quad UNA_{i-1} = \frac{\sum_{k=1}^{i-1} N_k}{\sum_{k=1}^{i-1} F_k}, \quad UN_{i-1} = \frac{N_{i-1}}{F_{i-1}} \right) \tag{3}$$

As shown in Equation (3), the buffer ratio adjustment cycle (i.e., F_i) for the i-th time period is determined by multiplying the ratio of the number of peers that have joined and left per time unit during the previous time period (i.e., UN_{i-1}) to the number of peers that have joined and left per time unit during all previous time periods (UNA_{i-1}) by average cycle during all previous time periods (i.e., AF_{i-1}). That is, F_i is obtained by reflecting the degree of ratio fluctuation in the previous time period in AF_{i-1}. Therefore, in our proposed adaptive buffering scheme, the buffer ratio adjustment cycle in each time period is dynamically determined according to the frequency of member change in each group.

4. Performance Evaluation

To show the effectiveness of our proposed adaptive buffering scheme in P2P live and time-shifted streaming systems, we performed extensive simulations using the PeerSim P2P simulator that supports a virtual P2P network environment. The ratio of the bandwidth between a peer and a router was set to 10–100 Mbps so that 100 Mbps, 50 Mbps, 20 Mbps, and 10 Mbps can be 10%, 50%, 20%, and 20%, respectively. The backbone bandwidth was set to 10 Gbps. The maximum number of peers to which a media server can transmit data directly and the maximum number of neighbors of each peer were set to 25 and 7, respectively. The inter-arrival and inter-leaving rates followed a Poisson distribution with a mean of 600 s. Each video had a 720-Kbps playback rate, and the size of each chunk was 30 KB. When a peer joined, it buffered at least 45 chunks of the requested video (corresponding to 15 s) to compensate for peer churn and rate fluctuations of video connections before initial playback. However, this does not mean that each peer must wait for 15 s. It could receive 45 chunks quickly if it worked in a better condition; for example, where the maximum possible number of neighbors are connected, the upload bandwidths of all neighbors are high, and network traffic is low. We also assumed that parameter α used to determine the buffer area ratio was 0.9. Table 2 shows the simulation parameter values used throughout this section.

Table 2. Simulation parameters.

Parameter	Default Value
Number of participating peers	2400/1800/1200
Bandwidth between router and peer	100 Mbps: 240/180/120 peers
	50 Mbps: 1200/900/600 peers
	20 Mbps: 480/360/240 peers
	10 Mbps: 480/360/240 peers
Backbone network bandwidth	10 Gbps
Maximum service capacity of media server	25 peers
Maximum number of neighbors of each peer	7
Average inter-arrival and inter-leaving rate	600 s
Video playback rate	720 Kbps
Chunk size	30 KB
Number of chunks per buffermap	1024/512/256
Number of buffered chunks required for initial playback	15 s (45 chunks)
α	0.9

To compare the performance of our proposed adaptive buffering scheme with the conventional fixed buffering scheme according to the number of peers, we varied the number of peers to 1200, 1800, and 2400. To show the performance of both schemes according to buffermap size, we also changed the numbers of chunks in a buffermap to 256, 512, and 1024.

On the other hand, to reflect the fact that many peers' requests are concentrated around the live broadcasting time and the others are dispersed over the entire playback period of each video due to time-shifted viewing, we distributed all peers into three types of playback zones, as shown in Figure 5. That is, 6000 s are divided into the following three zones: zone A within 60 s from the live broadcasting time, zone B between 60 and 660 s, and zone C between 660 and 6000 s. We assigned 50%, 40%, and 10% of all peers to zones A, B, and C, respectively, for all numbers of peers: 1200, 1800, and 2400.

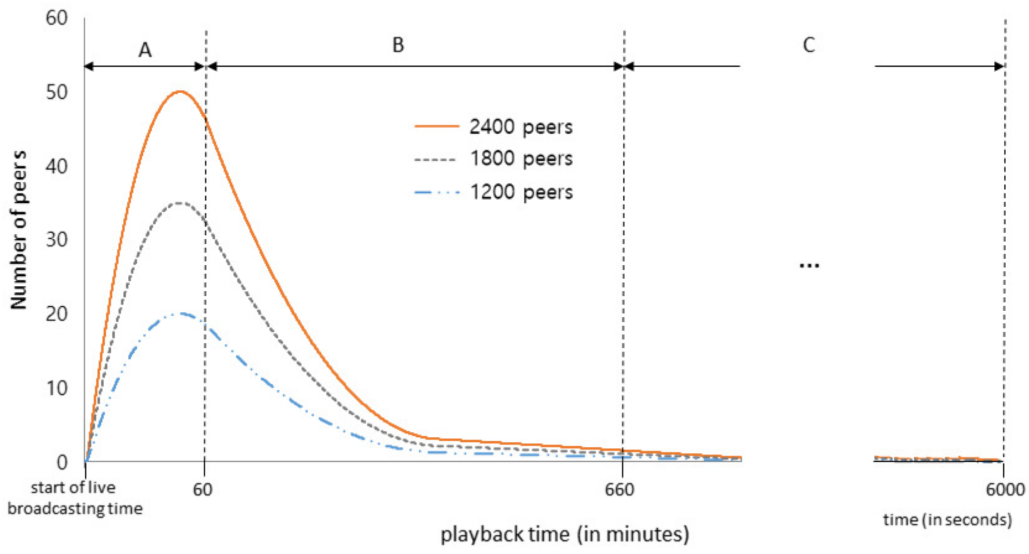

Figure 5. Assignment of peers to playback zones.

We compared the performance of our proposed adaptive buffering scheme with the conventional fixed buffering scheme in terms of startup delay, jitter ratio, and the ratio of necessary chunks in a buffermap. The ratio of necessary chunks in a buffermap represents the ratio of the number of chunks actually needed by a peer to the total number of chunks that are marked as available in the buffermaps received from neighbor peers.

4.1. Startup Delay

Figure 6 shows the startup delays of the adaptive buffering scheme and the fixed buffering scheme when the numbers of participating peers are 1200, 1800, and 2400. When a peer joins or changes its playback position by performing time-shifted functions, if its buffering period is not overlapped with those of any neighbor peers, it waits until at least one neighbor peer can transmit the data required for playback. The startup delays in the simulations include all such waiting times.

We can see that the startup delays of the adaptive buffering scheme are shorter than those of the fixed buffering scheme for all numbers of peers. That is, the startup delays of the adaptive buffering scheme are shorter by 17.1, 16.6, and 15.5 s when the numbers of peers are 1200, 1800, and 2400, respectively. In particular, when 1200 peers are participating, the startup delay of the adaptive buffering scheme is 47.2 s, while that of the fixed buffering scheme is 64.3 s, which is a 26.6% improvement. This is because the adaptive buffering scheme can considerably increase the degree of data duplication among peers by effectively adjusting the ratio of caching and prefetching areas according to the relative playback positions of each peer in a group. Therefore, the data required for initial playback can be received more quickly. On the contrary, in the fixed buffering scheme, since the ratio of caching and prefetching areas is fixed and the buffering status of other peers is not considered, there is no way to dynamically increase the overlapping degree in buffered data among peers.

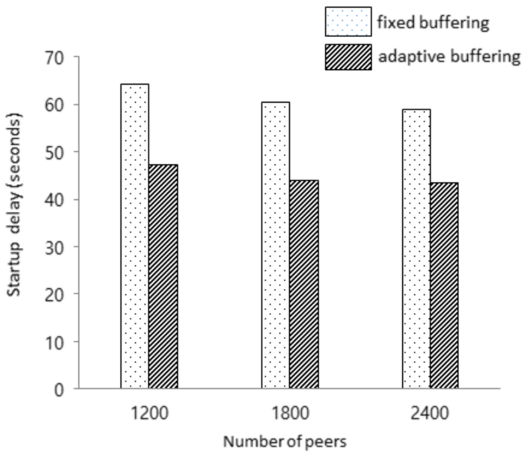

Figure 6. Startup delay according to the number of peers (number of chunks per buffermap: 512).

It can be also seen from Figure 6 that as the number of participating peers decreases, the startup delays of both schemes increase. This is because as the number of participating peers decreases, the number of neighbor peers to which each peer can connect also decreases. Thus, it takes longer for each peer to receive a sufficient amount of data to initiate playback.

Figure 7 shows the startup delays of the two buffering schemes when the buffermap size changes between 256, 512, and 1024 chunks. The startup delays of the adaptive buffering scheme are also shorter than those of the fixed buffering scheme for all buffermap sizes. The adaptive buffering scheme shows startup delays that are 17.7, 17.1, and 14.8 seconds shorter when the number of chunks in a buffermap is 256, 512, and 1024, respectively. Especially, when the buffermap size is 256 chunks, the startup delays of the adaptive buffering scheme and fixed buffering scheme are 48.2 s and 65.9 s, respectively. This performance improvement of 26.9% is also due to the increase in the degree of data duplication among peers.

Figure 7 also shows that as buffermap size increases, the difference in startup delay between the two schemes decreases slightly. This is because in the fixed buffering scheme, the amount of buffered data also increases with increase in buffer size, thereby increasing the amount of data to be shared among peers.

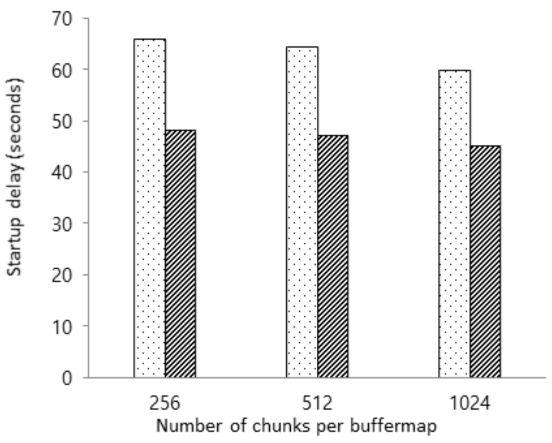

Figure 7. Startup delay according to the number of chunks per buffermap (number of peers: 1200).

4.2. *Jitter Ratio*

Figures 8 and 9 show the average jitter ratios of the two buffering schemes when varying the total number of peers and the number of chunks per buffermap. As expected, the jitter ratios of the adaptive buffering scheme are lower in all numbers of peers and chunks per buffermap. When the number of peers are 1200, 1800, and 2400, the jitter ratios of the adaptive buffering scheme are 32.5%, 32.4%, and 30.6% lower than those of the fixed buffering scheme, respectively. When the numbers of chunks per buffermap are 256, 512, and 1024, the adaptive buffering scheme achieves performance improvements of 42.1%, 32.5%, and 28.3%, respectively. The reason for this is that in the adaptive buffering scheme, each peer can provide relatively more data to others because peers can share a relatively larger amount of data with each other with increase in data duplication among peers.

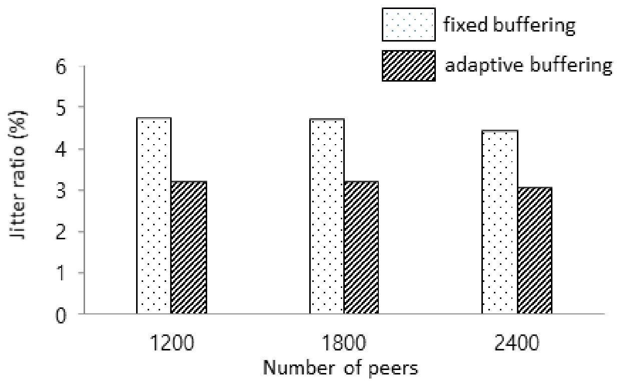

Figure 8. Jitter ratio according to the number of peers (number of chunks per buffermap: 512).

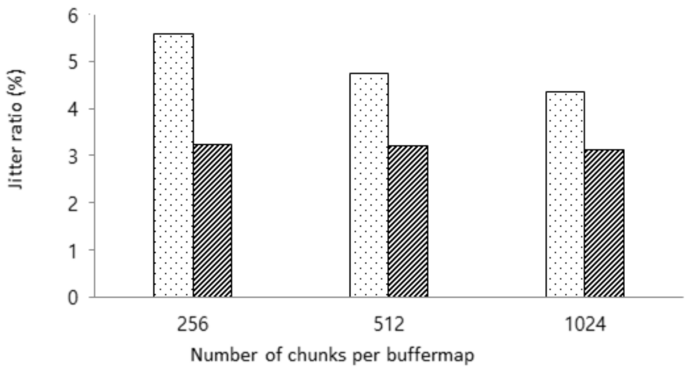

Figure 9. Jitter ratio according to the number of chunks per buffermap (number of peers: 1200).

It is noted that the difference in the jitter ratio between the two schemes increases as the buffermap size decreases. This is because even though buffer size is small, the adaptive buffering scheme can still increase the overlap in buffered data among peers by properly adjusting the buffer ratio of each peer.

4.3. *Ratio of Necessary Chunks in a Buffermap*

Figures 10 and 11 show the ratios of necessary chunks in a buffermap while changing the number of peers and the number of chunks per buffermap. The ratio of necessary chunks in a buffermap indicates the ratio of actually needed chunks among available ones in a buffermap. Therefore, the ratio of necessary chunks in a buffermap and the jitter ratio show contrary results. Thus, the adaptive buffering scheme in all numbers of peers and chunks per buffermap has a relatively higher ratio of necessary chunks in a buffermap. The adaptive buffering scheme shows 24.3%, 23.5%, and 22.6% higher ratios of necessary chunks per buffermap than the conventional scheme as the number of peers changes between 1200, 1800, and 2400. It also shows 24.6%, 24.3%, and 22.2% higher ratio as the buffermap size changes between 256, 512, and 1024 chunks.

Consequently, our simulation results show that our proposed adaptive buffering scheme significantly improves performance in all our simulation settings by effectively adjusting the ratio of caching and prefetching areas in the buffer of each peer.

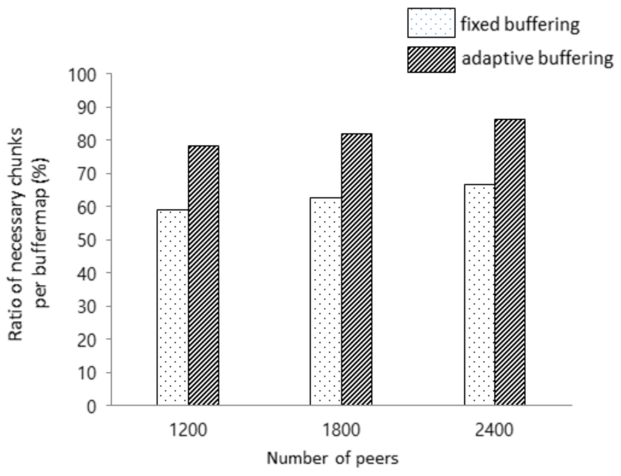

Figure 10. Ratio of necessary chunks in a buffermap according to the number of peers (number of chunks per buffermap: 512).

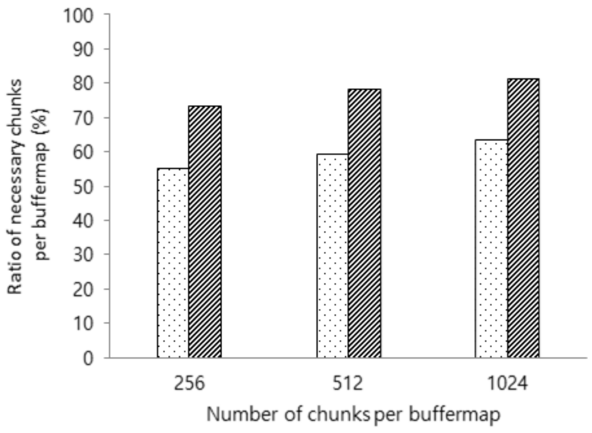

Figure 11. Ratio of necessary chunks in a buffermap according to the number of chunks per buffermap (number of peers: 1200).

5. Conclusions

In this paper, we proposed a novel buffering scheme to adaptively adjust the ratio of caching and prefetching areas in the buffer of each peer by considering the characteristics of P2P live streaming systems, where peers buffer only a small amount of data that are distributed widely. In our proposed scheme, the buffer ratio of each peer is determined according to the relative position of its playback time in a group after forming groups of peers according to the time periods of their playback positions. We increase the ratio of the caching area in the buffer of a peer as its playback time within a group moves earlier and increase the ratio of the prefetching area as the position moves later. We also showed that the adaptive buffering scheme can achieve considerable improvement in startup delay, average jitter ratio, and ratio of necessary chunks in a buffermap by increasing data duplication among peers.

Acknowledgments: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2016R1D1A1A09917396) and by the MSIP (Ministry of Science, ICT & Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2016-H8501-16-1006) supervised by the IITP (Institute for Information & communications Technology Promotion).

Author Contributions: Eunsam Kim and Taeyoung Kim conceived and designed the simulations; Taeyoung Kim performed the simulations; Eunsam Kim and Choonhwa Lee analyzed the data and wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sentinelli, A.; Marfia, G.; Gerla, M.; Kleinrock, L.; Tewari, S. Will IPTV Ride the Peer-to-Peer Stream? *IEEE Commun. Mag.* **2007**, *45*, 86–92.
2. Kim, D.; Kim, E.; Lee, C. Efficient peer-to-peer overlay networks for mobile IPTV services. *IEEE Trans. Consum. Electron.* **2010**, *56*, 2303–2309.
3. Lee, C.; Kim, S.; Kim, E. Expediting P2P Video Delivery through a Hybrid Push-Pull Protocol. *Adv. Electr. Comput. Eng.* **2015**, *15*, 3–8.
4. Zhang, X.; Liu, J.; Li, B.; Yum, T. CoolStreaming/DONet: A Data-driven Overlay Network for Peer-to-Peer Live Media Streaming. In Proceedings of the IEEE INFOCOM, Miami, FL, USA, 13–17 March 2005; pp. 2102–2111.
5. Liao, X.; Jin, H.; Liu, Y.; Ni, L.; Deng, D. AnySee: Scalable Live Streaming Service based on Inter-Overlay Optimization. In Proceedings of the IEEE INFOCOM, Barcelona, Spain, 23–29 April 2006; pp. 1663–1674.
6. Hei, X.; Liang, C.; Liang, J.; Liu, Y.; Ross, K. A Measurement Study of a Large-Scale P2P IPTV System. *IEEE Trans. Multimed.* **2007**, *9*, 1672–1687.
7. Li, Z.; Cao, J.; Chen, G. ContinuStreaming: Achieving High Playback Continuity of Gossip-based P2P Streaming. In Proceedings of the IEEE IPDPS, Miami, FL, USA, 14–18 April 2008; pp. 1–12.
8. Hei, X.; Liu, Y.; Ross, K. IPTV over P2P Streaming Networks: The Mesh-Pull Approach. *IEEE Commun. Mag.* **2008**, *46*, 86–92.
9. Do, T.; Hua, K.; Tantaoui, M. P2VoD: Providing Fault Tolerant Video-on-Demand Streaming in Peer-to-Peer Environment. In Proceedings of the IEEE International Conference on Communications, Paris, France, 20–24 June 2004; pp. 1467–1472.
10. Vratonjic, N.; Gupta, P.; Knezevic, N.; Kostic, D.; Rowstron, A. Enabling DVD-like features in P2P video-on-demand systems. In Proceedings of the Peer-to-peer streaming and IP-TV, Kyoto, Japan, 27–31 August 2007; pp. 329–334.
11. Cheng, B.; Stein, L.; Jin, H.; Liao, X.; Zhang, Z. GridCast: Improving Peer Sharing for P2P VoD. *ACM Trans. Multimed. Comput. Commun. Appl.* **2008**, *4*, 1–31.
12. Cheng, B.; Stein, L.; Jin, H.; Zhang, Z. A framework for lazy replication in P2P VoD. In Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video, Braunschweig, Germany, 28–30 May 2008; pp. 93–98.
13. He, Y.; Shen, G.; Xiong, Y.; Guan, L. Optimal Prefetching Scheme in P2P VoD Applications with Guided Seeks. *IEEE Trans. Multimed.* **2009**, *11*, 138–151.
14. Bartolini, N.; Nikoletseas, S.; Sinha, P.; Cardelliniand, V.; Mahanti, A. COCONET: Co-operative Cache Driven Overlay NETwork for P2P VOD Streaming. In Proceedings of the International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, Las Palmas, Spain, 23–25 November 2009; pp. 52–68.
15. Tang, H.S.; Chan, S.H.G.; Li, H. Optimize Segment Caching for Peer-to-Peer On-demand streaming. In Proceedings of the 2009 IEEE International Conference on Multimedia and Expo (ICME), New York, NY, USA, 28 June–3 July 2009; pp. 810–813.
16. Fujimoto, T.; Endo, R.; Matsumoto, K.; Shigeno, H. Video-Popularity-based Caching Scheme for P2P Video-on-Demand Streaming. In Proceedings of the 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications, Biopolis, Singapore, 22–25 March 2011; pp. 748–755.
17. Wu, W.; Lui, J. Exploring the Optimal Replication Strategy in P2P-VoD Systems: Characterization and Evaluation. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *23*, 1492–1503.

18. Guo, P.H.; Yang, Y.; Li, X.Y. A P2P streaming service architecture with distributed caching. *J. ZHEJIANG Univ. Sci. A* **2007**, *8*, 605–614.
19. Wauters, T.; Meerssche, W.; Turck, F.; Dhoedt, B.; Demeester, P.; Van Caenegemand, T.; Six, E. Co-operative Proxy Caching Algorithms for Time-Shifted IPTV Services. In Proceedings of the EUROMICRO Software Engineering and Advanced Applications, Cavtat, Croatia, 29 August–1 September 2006; pp. 379–386.
20. Abboud, O.; Pussep, K.; Kovacevic, A.; Mohr, K.; Kaune, S.; Steinmetz, R. Enabling resilient P2P video streaming: Survey and analysis. *Multimed. Syst.* **2011**, *17*, 177–197.
21. Passarella, A. A survey on content-centric technologies for the current Internet: CDN and P2P solutions. *Comput. Commun.* **2012**, *35*, 1–32.
22. Markakis, E.; Pallis, E.; Skianis, H.; Zaxaropoulos, V. Dynamic TCP window size adaptation for improving network performance of p2p-aware interactive broadcasting environments. *J. Commun. Comput.* **2012**, *9*, 1395–1403.
23. Markakis, E.; Negru, D.; Bruneau-Queyreix, J.; Pallis, E.; Mastorakis, G.; Mavromoustakis, C. A P2P Home-Box Overlay for Efficient Content Distribution. In *Emerging Innovations in Wireless Networks and Broadband Technologies*; IGI Global: Hershey, PA, USA, 2016; pp. 199–220.
24. Adzic, V.; Kalva, H.; Furht, B. A survey of multimedia content adaptation for mobile devices. *Multimed. Tools Appl.* **2011**, *51*, 379–396.
25. Letian, R.; Burnett, I. Facilitating Universal Multimedia Adaptation (UMA) in a Heterogeneous Peer-to-Peer Network. In Proceedings of the IEEE Conference on Automated Production of Cross Media Content for Multi-Channel Distribution, Leeds, UK, 13–15 December 2006.
26. Tran, D.; Huaand, K.; Do, T. ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming. In Proceedings of the IEEE INFOCOM, San Francisco, CA, USA, 30–31 March 2003; pp. 1283–1292.
27. Castro, M.; Druschel, P.; Kermarrec, A.; Nandi, A.; Rowstronand, A.; Singh, A. SplitStream: High-Bandwidth Multicast in Cooperative Environments. In Proceedings of the ACM Symposium on Operating Systems Principles, Bolton Landing, NY, USA, 19–22 October 2003; pp. 298–313.
28. Mondal, A.; Lifuand, Y.; Kitsuregawa, M. P2PR-Tree: An R-Tree-based Spatial Index for Peer-to-Peer Environments. In Proceedings of the International Conference on Extending Database Technology, Heraklion, Greece, 14–18 March 2004; pp. 516–525.
29. Liu, Z.; Shen, Y.; Rossand, K.; Panwar, S. LayerP2P: Using Layered Video Chunks in P2P Live Streaming. *IEEE Trans. Multimed.* **2009**, *11*, 1340–1352.
30. Chen, Y.; Chen, C.; Li, C. Measurement Study of Cache Rejection in P2P Live Streaming System. In Proceedings of the IEEE Conference on Distributed Computing Systems, Beijing, China, 17–20 June 2008; pp. 12–17.