Multimed Tools Appl (2017) 76:2557–2568
DOI 10.1007/s11042-016-3249-x

An efficient hybrid push-pull methodology for peer-to-peer video live streaming system on mobile broadcasting social media

Ha Thi Thu Tran[1] · **Yonggwan Won**[1] · **Jinsul Kim**[1]

Received: 15 August 2014 / Revised: 29 October 2015 / Accepted: 5 January 2016 /
Published online: 18 January 2016

Abstract With the rapid growth of wireless communication technology, the availability of highly flexible and video-friendly mobile terminal platforms (such as smartphones and tablets), the emergence of major video content providers (like YouTube, Ustream, and PPTV, which provide a large catalog of attractive contents), Peer-to-Peer (P2P) live video streaming over the wireless and Internet is becoming more and more attractive to users. One of the main challenges is to provide a good quality of service though the dynamic behavior of the network. Traditionally, tree-based model uses a push method, that broadcaster transfers data to other users. This model has low start-up delay. However, there are two main problems in this method: if the bandwidth of an internal node is low, children nodes may lose data and when an internal node failure, other nodes can't receive data until completing the recovery of the tree. On the other hand, mesh-based model uses a pull method, has low bandwidth of a neighbor node by pulling necessary data from a number of neighbor nodes. However, mesh-based model requires large buffers to support pull data from neighbor peers and there is an adjustment between minimum delay by sending pull request and overhead of whole system. So, both models have their own strengths and weaknesses. This paper proposes a new hybrid push-pull live P2P video streaming protocol called MobileCast that combines the benefits of pull and push mechanisms for live video delivery. We present new topology for P2P network with more stable and provide better video streaming quality. Our main goal is to minimize the network end-to-end delay, startup time, overhead, packet loss compared to the pure mesh networks, pure tree networks and provide a good quality of service though the dynamic behavior of the network.

✉ Jinsul Kim
jsworld@jnu.ac.kr

Ha Thi Thu Tran
thuhabkhn@gmail.com

Yonggwan Won
ykwon@jnu.ac.kr

[1] School of Electronics and Computer Engineering, Chonnam National University, Gwangju, Korea

Keywords Peer–to–peer · Live video streaming · File sharing · Broadcasting · Pull and push

1 Introduction

The rapid development of smart phone has created an opportunity for mobile multimedia services for mobile users. P2P live video streaming is becoming an increasingly popular technology. Streaming generally is a method for intelligent broadcasting of data on the mobile phone, it differs from conventional multimedia services because it isn't necessary to wait for the end of downloading video and able to start playing back. P2P live video streaming is a technology that allows a user is called broadcaster generates a video that is transmitted to other users in real-time. But P2P live video streaming system is facing with the dynamic join/leave of users, a peer may join or leave anytime so the structure of the P2P streaming must be able to deal with all type of dynamic changes, also mobile devices have limitations: processor capacity, bandwidth, memory, and battery. [3, 5] propose an idea in which mobile user is both capable of broadcasting and watching live video at the same time using BitTorrent live streaming, the quality and reliability of the broadcast actually improves every time a new viewer joins, since every new viewer becomes a miniature broadcaster and amplifies the stream.

In [11] showed that P2P live streaming network exist two types of users: streaming users and storage users. The streaming users who use mobile devices with 3G/4G connection expect to watch the live video immediately, and the storage users who use PC with wired Internet will download and then watch the video later. When the storage users start to watch, they can either watch from the beginning or watch the current live stream, assume that the live broadcast is not finished. The streaming users may stop watching live video after a while if they find the video is out of their interest. Users leaving causes dynamic and affect the data delivery. On the other hand, the storage users that are downloading the video do not have the concern of interest and playback quality, until they start to watch the video. Hence, the storage users are relatively more stable, so we call the storage users are storage nodes and streaming users are streaming nodes.

This paper proposes method to solve how to schedule the video content delivery and construct network to improve the overall performance of the system. We introduce new architecture system design for P2P live video streaming that combines the advantages of pull and push methods for broadcasting live video. Nodes are constructed into sub-trees, a node belongs to only one sub-tree. Each node also has links to other nodes of other sub-trees (pull-link). Video is split to sub-streams. Each sub-stream is delivered through a sub-tree based on a push mechanism (push-link). Each node will receive from its parent at least one sub-stream then, node pulls other sub-streams from other nodes to improve the quality of video. Also, we design network topology with storage nodes are close to the broadcaster, because they are more stable than streaming node which can leave system anytime. Thus, the storage nodes are relatively more stable, so we design the storage nodes are close to the broadcaster. We call the storage users are storage nodes and streaming users are streaming nodes for design new network topology. Structure of remain paper as following: part 2: related work, part 3: proposed idea and system design, part 4: performance evaluation, and part 5: conclusion.

2 Related works

According to P2P overlay topology, P2P streaming methods can be classified into two categories: tree-based (e.g., ChunkySpread- is unstructured, using multiple trees to balance load among nodes, it also reacts quickly to membership changes and scales well with low overhead [9]) and mesh-based (e.g., CoolStreaming- every node periodically exchanges data availability information with a set of partners, and retrieves unavailable data from partners [12]).

Tree- based model has content flows from root to children along the tree, node failures affect a complete sub-tree and long recovery time. Push approach is applied to P2P tree topology where all nodes in the system form a tree structure as shown in the left side of Fig. 1. Parent nodes push video frames to their child nodes. This approach is reported to achieve lower delay. The scheduling in a tree-based streaming system includes: choose which video frame to push to child node, choose which child node to push selected frame based on the tree structure [2, 7].

Besides that, the scheduling in a mesh-based streaming system includes: choose which video frame to be request based on own buffer map, choose which peer to send frame request based on the neighbor's buffer map, resend request to partners for necessary frame [1, 8]. Mesh-based has nodes exchanges data availability information with neighbor nodes, resilient to node failure, high control overhead, meta-data exchange consumes bandwidth, longer delay for downloading each chunk.

As shown in the right side of Fig. 1, when a peer joins the network, firstly, it contacts a node is called tracker that has information about the existing peers in the network. The tracker returns the list of neighbors that are close to new peer [6]. After receiving the neighbors' address, the newly joined peer contacts each of them and sends a join request message to inform them that it wants to make a connection. On the other side, if the peer has a free position for the nodes request, it sends a join respond message to the new peer. When the newly joined peer receives the accept message, it replies with a join ACK message to confirm the neighbor connection between itself and then establishing the connection between two nodes [4, 10].

It is known that a tree-based with data push is more efficient than a mesh-based with data pull, but maintaining the tree overlay with node churns is a difficult task.

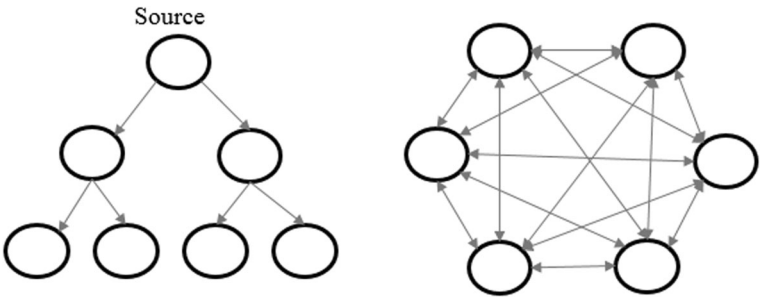

Fig. 1 Tree-based topology and Mesh-based topology

3 Proposed idea and system design

3.1 Proposed idea

In MobileCast, we consider a local network consisting of four users: A, B, C, and D. User A captures live video and then broadcasting this video on P2P network as shown in Fig. 2. We assume that this video is split into three segments s_1, s_2, and s_3. User A pushes three segments to users B, C, and D with respectively. After finishing receiving each segment s, then B, C, or D send requests to each other for remain segments of video which user A broadcasted, using pull requests. For example, user B receives segment s_1 from user A and then it sends pull request to user C and D to get segment s_2 and s_3 of video. Now, user B is watching a segment s_1 which receives from user A and also broadcasting this segment for C, D. At this time, user B is also getting segments s_2, s_3 from C, D and continues watching whole live video. We can extend this scenario with many users, building tree overlay with a number of sub-streams. In each sub-stream, push method is used like the case user A pushes data to user B, C and D. Besides that, in each level, pull method is used like the case user B pull data from user C and D. This is key idea for our method.

It is known that, the streaming users who use mobile devices with 3G/4G connection and the storage users who use PC with wired Internet. The streaming users may stop watching live video after a while if they find the video is out of their interest. Users leaving causes dynamic and can affect the data delivery. Besides, the storage users do not have the concern of interest and playback quality. Thus, the storage nodes are relatively more stable, so we design the storage nodes are close to the broadcaster. We call the storage users are storage nodes and streaming users are streaming nodes for design new network topology.

Figure 3 shows the organization of two types of nodes (storage node and streaming node) in the overlay tree. For this case, we can split overlay tree into three small sub-trees. User are organized in separate sub-trees, except the broadcaster. That means a node belongs to only one

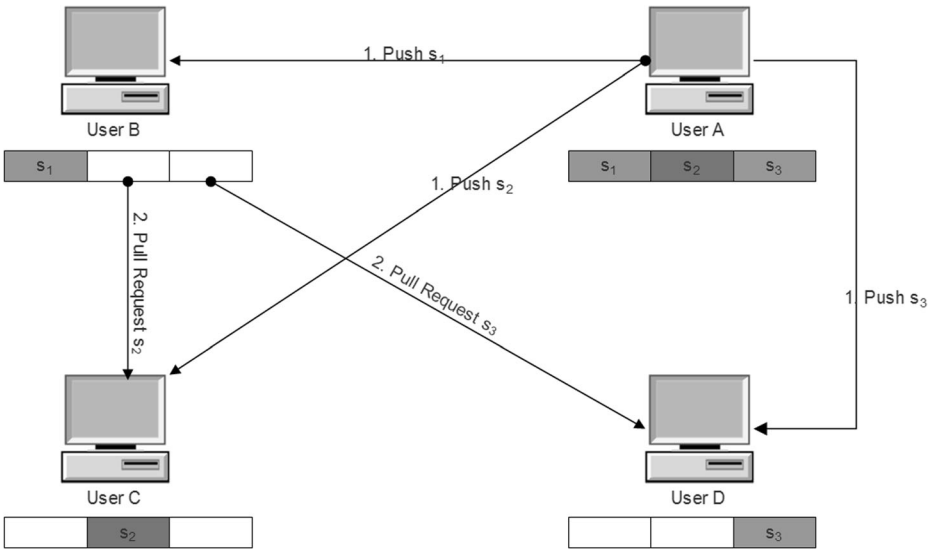

Fig. 2 Live video streaming system

sub-tree. Each node also has links to other nodes of other sub-trees. Each node maintains two kinds of connections: connection belonging to a sub-tree (push link) and connections of nodes in different sub-trees (pull link). Broadcaster sends live video to each sub-tree using push mechanism. Live video is divided into many chunks (is called sub-stream). Each node will receive from its parent at least one sub-stream in the pushing phase. Then, node pulls other sub-streams from other nodes in same level to get completed video and improve the quality of service.

According to the peer-to-peer protocol, when a node wishes to join the network to streaming video, it transmit a request to tracker for information about nodes in the P2P network as shown in flowchart Fig. 4. Then, node will create connection with nodes who are streaming video.

3.2 Proposed idea

Using the following metrics which together reflect the quality of service experienced by end user: start-up delay (it is the time taken by a node between its request of joining the overlay and receiving enough data blocks to start playing back), data loss rate (it is defined as the fraction of the data that have missed their playback deadlines), control overhead (it is size of the control messages sent by tree node).

As show in Fig. 5, at the push phase, broadcaster pushes three parts (3 sub-streams) to nodes at level 1. These nodes continue to push these parts to other child nodes in their sub-trees. As the result, through the pushing phase, nodes have minimum data (e.g., all nodes at level 2 of sub-stream already have data of sub-stream 1) required to display. At the pull phase, a node will send pull request to other node with same level to receive remain parts of video (e.g., at the level 1, node 1 sends pull request to node 2 and 3). So, streaming data is distributed to every node in the overlay network by both pushing and pulling methods.

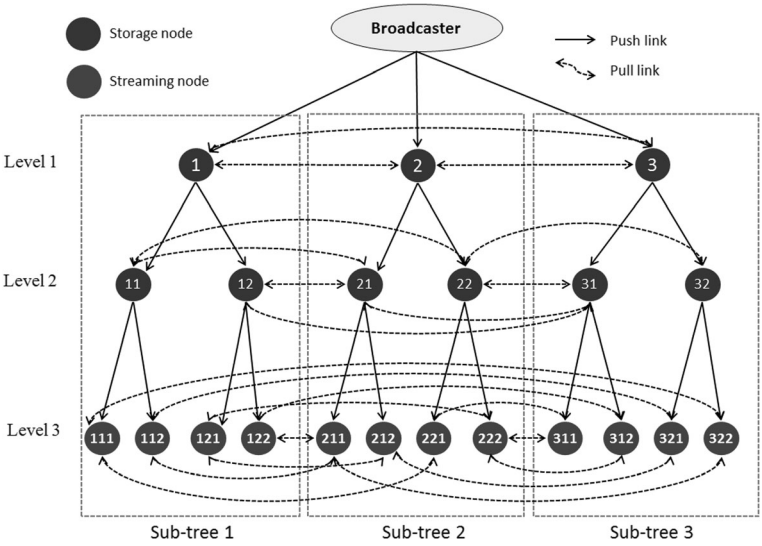

Fig. 3 MobileCast Overlay Structure

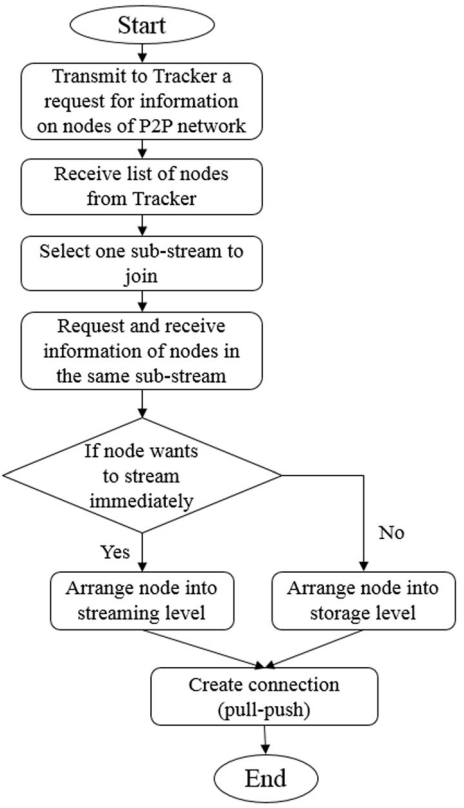

Fig. 4 Flowchart of processing for a node joining P2P network

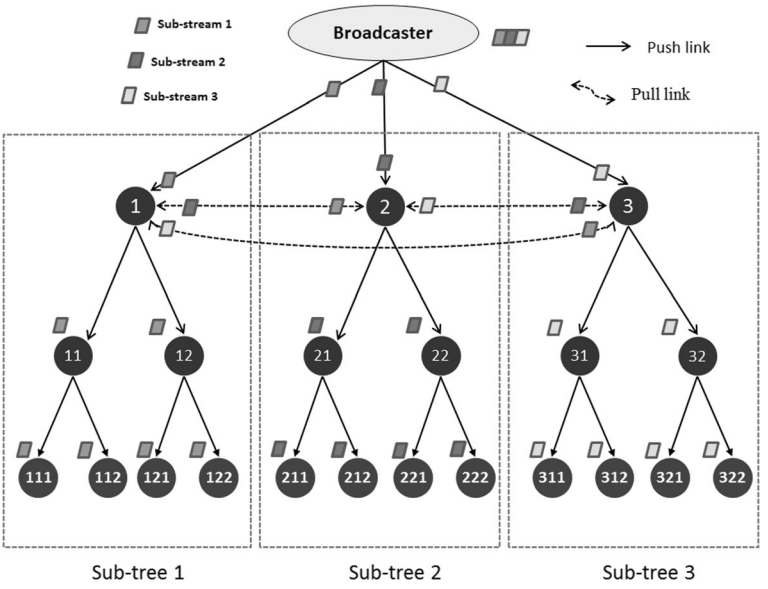

Fig. 5 Pushing data to sub-trees

In Fig. 6, describes processing of a node create video and distribute each part of video to other nodes. Figure 7 shows the pulling phase, in which nodes in sub-tree 1 (e.g., node receive 1st sub-stream in pushing phase) establish pull requests to nodes in other sub-trees in the same level to get 2nd and 3rd sub-streams.

The user requires pulling requests of other users which belong to other sub-trees. If this user didn't respond pulling requests of other users, it will inform other users about its rejection. These users must look up other users that can send data to them. If it doesn't send data to any user in a sub-tree, it may not able to pull data from any user in other sub-tree.

Figure 8 shows the pulling phase, in which nodes in sub-tree 1 (e.g., node receive 1st sub-stream in pushing phase) establish pull requests to nodes in other sub-trees in the same level to get 2nd and 3rd sub-streams. The user requires pulling requests of other users which belong to other sub-trees. If this user didn't respond pulling requests of other users, it will inform other users about its rejection. These users must look up other users that can send data to them. If it doesn't send data to any user in a sub-tree, it may not able to pull data from any

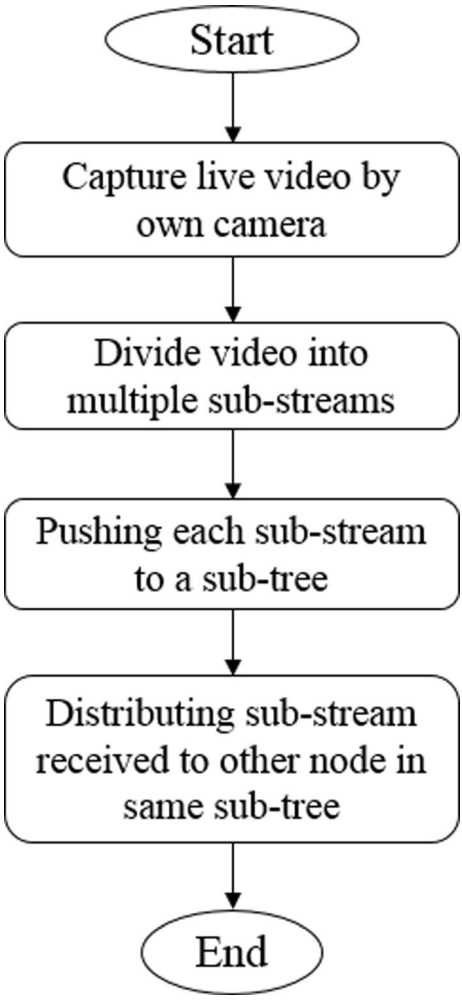

Fig. 6 Flowchart of processing of a node distributing content received (push data)

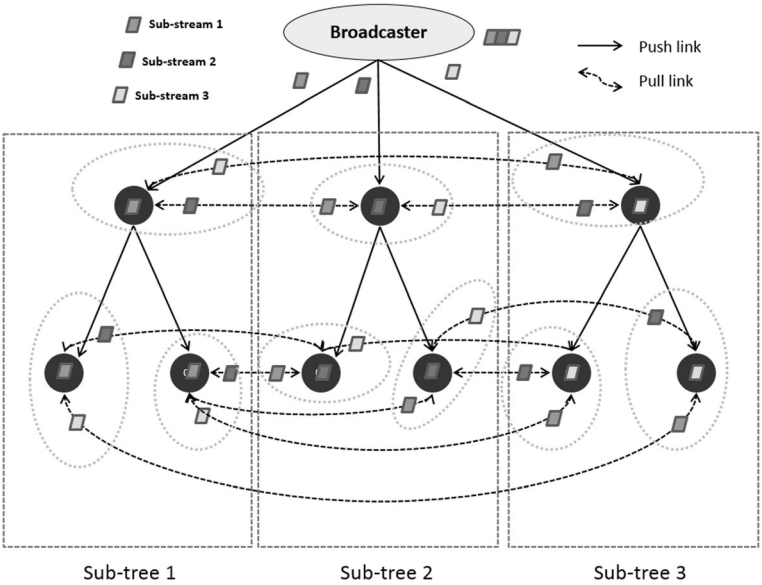

Fig. 7 Node pull missing data from node in other sub-trees

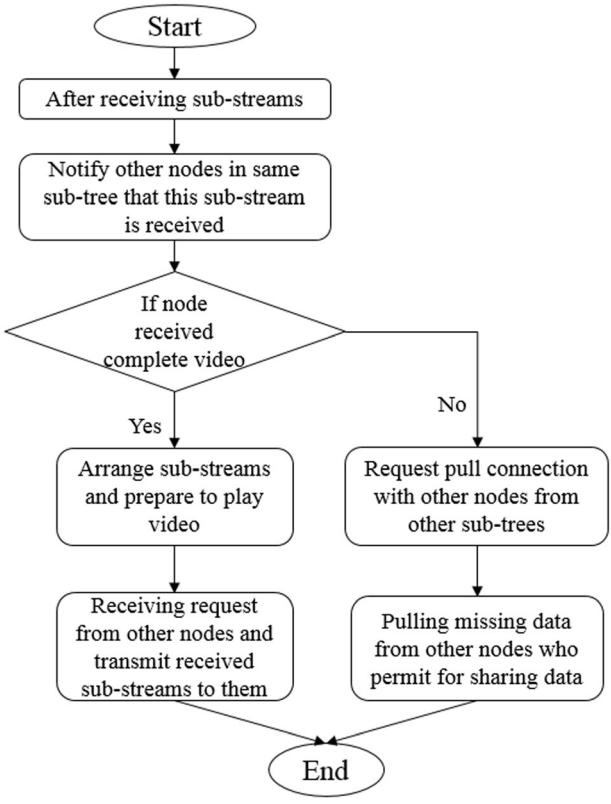

Fig. 8 Flowchart of processing of node to receive streaming content (pull data)

user in other sub-tree. When a node in a sub-tree receives a sub-stream, it will be flow schedule shown in Fig. 8.

4 Performance evaluation

We present our evaluation for MobileCast solution. In evaluation, we use the following typical metrics: startup delay, packet loss, control overhead to compare with Chunkyspread [9] and CoolStreaming [12] overlays. ChunkySpread is a typical tree-based overlay network. Chunkyspread is unstructured, using multiple trees to balance load among nodes. It also reacts quickly with low overhead. On the other hand, CoolStreaming is a mesh-based overlay network for live video streaming, in which every node periodically exchanges data availability information with a set of partners, and retrieves unavailable data from partners. The design is not only efficient but also robust and resilient.

Our simulation set up with 100 nodes using NS2 in Ubuntu environment. The playback will not start until the user has obtained sufficient data (at least 10 s of video data). We set up the simulation 10 times for each overlay (tree-based, mesh-based and hybrid push-pull) to get the average results. The session length is set to 180 s. At the beginning 60 nodes are storage nodes, a storage node exchange to streaming nodes with a probability of at time. A streaming node leaves the system with a probability of at time.

We obtained some results to compare startup delay, packet loss rate and control overhead of three types of solution as following.

4.1 Startup delay time

For the beginning, the mesh-based solution (CoolStreaming) performs worst, because it needs a longer time to search and request for neighbor peers. The pure tree-based solution (Chunkyspread) performance took only 20 s to startup. Even though, our solution need to aware of the coexistence of the two types of nodes and explicitly prioritizes the service to the streaming nodes but at the beginning MobileCast constructs as tree overlay network. So, nodes need shorter time to startup than the CoolStreaming and almost similar with Chunkyspread startup time.

4.2 Packet loss rate

CoolStreaming with the mesh-based solution performs the best, because it is pull-based and thus is resilient to the node dynamics. On the other hand, Chunkyspread with the pure tree-based solution performs the worst, because the tree overlay is interrupted to suffer from the node dynamics. Our solution combines pull-based and push-based, it takes advantage of both solutions, so it performs much better than the pure tree-based solution and little less than mesh-based solution. By using hybrid of overlay structure, MobileCast can achieve the performance of the mesh-based solution.

4.3 Control overhead

From experiment, with data pull, the mesh-based solution suffers from much higher overhead than the other solution because Peers need a lot of control messages to request and receive

packet from each other. So, mesh-based spend more overhead to maintain network topology. While MobileCast solution has slightly less overhead. This is because two reasons: from begin design, MobileCast is the tree overlay, and the nodes that are close to the root are less dynamic; therefore, the nodes spend less overhead on maintaining the overlay when a node want to leave network.

5 Conclusion

This paper presents MobileCast- a new hybrid push-pull live P2P video streaming protocol, that combines the benefits of pull and push mechanisms for live video delivery and also MobileCast recognized existence of the two types of users for video streaming - there are wired Internet users and wireless mobile users. Specifically, internet users have better network connection, and thus they should be considered to be located close to the broadcaster. Moreover, since mobile users are usually charged for data usage, such users are not always suitable for forwarding data. These requirements call for suitable design of the architecture as shown in this paper with new topology for P2P network with more stable, minimize the network startup delay, packet loss and control overhead compare to the pure mesh networks, tree networks, and provide better video streaming quality. MobileCast successfully takes advantage of the coexistence of live streaming and storage sharing, providing better scalability, robustness, and streaming quality.

Acknowledgments This research was supported by Chonnam National University and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2013R1A1A2013740) and was partially supported by the IT R&D program of MSIP(Ministry of Science, ICT and Future Planning) / IITP(Institute for Information & Communications Technology Promotion) [12221-14-1001, Next Generation Network Computing Platform Testbed].

References

1. Chang H, Jamin S, Wang W (2011) Live streaming with receiver-based peer-division multiplexing. IEEE/ACM Trans Networking 19(1):55–68
2. Cheng X, Liu J (2012) Exploring interest correlation for peer-to-peer socialized video sharing. ACM Trans Multimed Comput Commun Appl 8, no.1, article 5
3. Cho C, Tran-Thi-Thu H, Park S, Kim J (2014) An efficient P2P-based mobile social media delivery for real-time mobilecast. Int J Softw Eng Appl 8(2):151–158
4. Fan B, John C, Lui S, Chiu D-M (2009) The design trade-offs of BitTorrent-like file sharing protocols. IEEE/ACM Trans Networking 17(2):365–376
5. Ha TTT, Won Y, Kim J (2014) Topology and architecture design for peer to peer video live streaming system on mobile broadcasting social media. International Conference on Information Science and Applications (ICISA)
6. Liu B, Cui Y, Lu Y, Xue Y (2009) Locality-awareness in BitTorrent-like P2P applications. IEEE Trans Multimedia 11(3):361–371
7. Magharei N, Rejaie R (2009) PRIME: peer-to-peer receiver-driven mesh-based streaming. IEEE/ACM Trans Networking 17(4):1052–1065
8. Sanna M, Izquierdo E (2013) Live scalable video streaming on peer-to-peer overlays with network coding. IEEE Lat Am Trans 11(3):962–968
9. Venkataraman V, Yoshida K, Francis P (2006) Chunkyspread: heterogeneous unstructured tree-based peer-to-peer multicast. In: Proc. IEEE ICNP

10. Wu D, Liang Y, He J, Hei X (2013) Balancing performance and fairness in P2P live video systems. IEEE Trans Circuits Syst Video Technol 23(6):1029–1039
11. Xu C, Liu J, Wang H, Wang C (2012) Coordinate live streaming and storage sharing for social media content distribution. IEEE Trans Multimedia 14(6):1558–1565
12. Zhang X, Liu J, Li B, Yum T-SP (2005) Coolstreaming/DONet: adata-driven overlay network for peer-to-Peer live media streaming. In: Proc. INFOCOM

Ha Thi Thu Tran is currently a M.S candidate at Smart Mobile and Media Computing Laboratory, School of Electronics and Computer Engineering, Chonnam National University, South Korea. She received B.S degree from the School of Electronics and Telecommunications, HaNoi University of Science and Technology, VietNam in 2011. She was a solution engineer at Asian Communication Solution, JSC in 2012. Her major interests are in the research areas of Mobile Cloud Computing, Next Generation of Mobile Platform, Mobile Operating System, and Peer-to-Peer Network.

Yonggwan Won He received his Ph.D and M.Sc in Electrical and Computer Engineering from the University of Missouri-Columbia. He has worked for Korean Army Headquarters and worked with GoldStar Telecommunication Ltd in 1987. Since 1988, he has conducted research in computer vision, image processing, pattern recognition, neural network, mathematic morphology, nonlinear filtering, fuzzy logic, handwritten digit recognition and automatic target recognition. He is now working in Department of Electronics and Computer Engineering, Chonnam National University as a professor. He is also the Director of Korea BIO-IT Foundry Center – Gwangju and Head of Special Research Group of IT Fusion Medical Systems.

Jinsul Kim received the B.S. Degree in computer science from University of Utah, Salt Lake City, Utah, USA, in 2001, and the M.S. and Ph.D degrees in digital media engineering, department of information and communications from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2005 and 2008. He worked as a researcher in IPTV Infrastructure Technology Research Laboratory, Broadcasting/Telecommunications Convergence Research Division, Electronics and Telecommunications Research Institute (ETRI), Daejeon, Korea from 2005 to 2008. He worked as a professor in Korea Nazarene University, Chon-an, Korea from 2009 to 2011. Currently, he is a professor in Chonnam National University, Gwangju, Korea. He has been invited reviewer for IEEE Trans. Multimedia since 2008. His research interests include QoS/QoE, Measurement/ Management, IPTV, Mobile IPTV, Smart TV, Multimedia Communication and Digital Media Arts.