Design a Hypercube-Tree Structure for Peer-to-Peer Streaming

Yu-Wei Chen and Ming-Cheng Lu

***Abstract*—In a peer-to-peer environment, how to effectively use network topology to provide services which prevent interruption is the major challenge. However, both tree-based and mesh-based structures are used to solve this, but they also have some drawbacks. In this paper, we propose a hypercube tree structure, referred to as HTS, in order to overcome drawbacks. In the proposed scheme, the hybrid structure consists of mesh-based and tree-based structure, and the main idea of mesh structure is to use eight nodes to construct a cube that we can regard it as a super node, and construct a hypercube tree. We hope to combine tree-based and mesh-based structures to avoid interruption that even if an interruption is generated, we can find other siblings to provide services.**

***Index Terms*—Peer-to-peer network, streaming, hypercube, tree structure, mesh structure**

I. Introduction

In recent years, peer-to-peer (P2P) technology has been becoming more and more popular, its main concept is that each node interconnects and operates as both client and server. Due to nodes increasing, the demand of network bandwidth and server load have been increasing, therefore, advantage brought by reduction of server load with using nodes resources is a common method.

P2P technology now has been a variety of applications on network, such as: (1) File sharing – a big part with Bit Torrent [1], for movies, music, software, documents and other files exchanging, (2)Streaming for audio and video sharing - PPTV [2], PPStream [3] and other software plays video online.

In P2P streaming, there are many approaches proposed such as: tree-based, mesh-based and hybrid-based. In the tree-based, nodes are connected to form a tree-structure, but any two nodes only exists one transmission path. In mesh-based, each node has two or more transmission paths to choose, and hybrid-based is a combination of tree-based and mesh-based approaches.

The above methods have their advantages and disadvantages. In the tree-based approach, the node's join/ leave behavior is very complex and difficult, because of in the P2P network, node is full of uncertainties, and may leave or interrupt at any time, once a node leaves, it may destroy the entire transmission path. For improving drawbacks of tree-based approach, past researches proposed some solutions such as: NICE [4], CoopNet [5], ZIGZAG [6], P2Cast [7], P2VOD [8], DircectStream [9].

Manuscript received March 12, 2012; revised May 5, 2012.
The authors are with Graduate Institute of Information and Logistics Management, National Taipei University of Technology, Taipei, Taiwan (e-mail: t9938006@ntut.edu.tw, ywchen@ntut.edu.tw).

In mesh-based approach, it has good fault-tolerant, once any transmission path is interrupted, it can be transmitted through other paths, but the network traffic will be raised, there has well-known approaches such as: CoolStreaming [10], PROMISE [11], Chainsaw [12], PRIME [13], VMesh [14].

HON [15], mTreeBone [16], BBTU [17], HyPo [18], and ToMo [19] are hybrid-based approaches in order to enhance transmission efficiency. In this paper, we have proposed a Hypercube-Tree structure of P2P streaming, it contains tree-based and mesh-based approaches in order to enhance robustness connection between nodes. We not only use parent node to establish connection with its child node, but also parent node's three sibling nodes to establish backup link with child node's three sibling nodes. That means, after an interruption is generated nodes finds other nodes to connect.

The rest of this paper is organized as follows: We introduce related works in Section 2. And in Section 3, we present HTS. In Section 4, we present the conclusion of this paper.

II. Related Work

A. P2Cast

In P2 Cast [7], it mainly focus on live streaming, Figure 1 shows an example of P2 Cast, nodes are clustered according to their arrival time and construct an application multicast tree, each node will received beginning of the part of video segments, when new node joins, if they do not receive initial video segments from the server, it can be patched from its parent's.

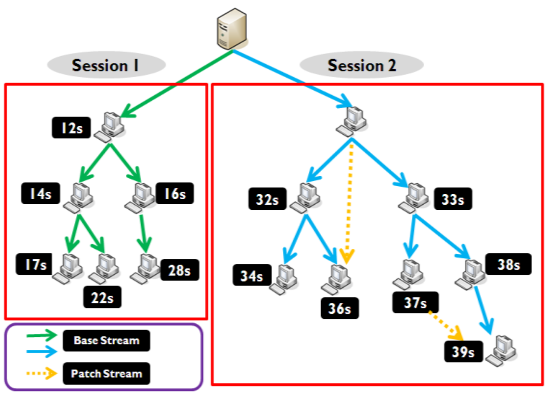

Fig. 1. P2Cast example

B. VMesh

In VMesh [14], it's a mesh-based structure, Figure 2 shows an example of Vmesh, videos are divided into segments and stored in the video server, each node will catch some segments according to their buffer size. Besides, each node uses distributed hash table and list to know the segments, in order to improve the segment supply. So if peer needs specific segments, it can find quickly, furthermore, it supports efficient interactive commands, like VCR.

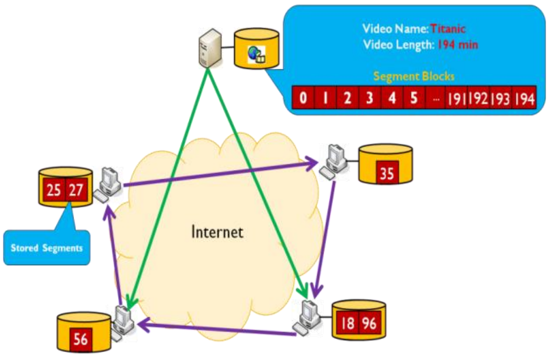

Fig. 2. VMesh example

C. HRT

In HRT [20], it combines ring-based and tree-based structures. It has some features: (1) It just needs a few messages to add or remove a peer; (2) It requires short recovery time without need of tree splitting or merging when nodes leave/disrupt.

Figure 3 shows an example of HRT, when nodes (node 1-node 6) joined, they are connected to a ring-base structure, and they have the same parent node (server), the followings are according to a breadth-first method (node 7-node 11 are constructing, their parent is node 1), then establish a backup link to enhance the system robustness (node 2 connects with node 12), when a parent node departs/fails, it uses the backup link to provide service to those node which has no parent node.

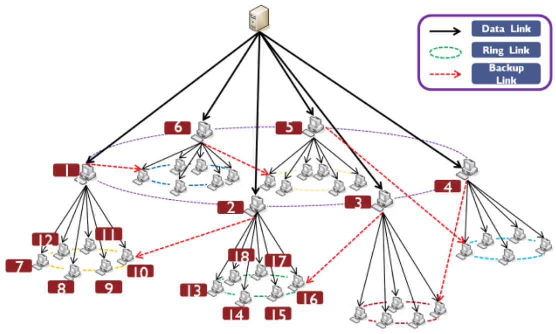

Fig. 3. HRT example

III. Hypercube Tree Structure

A. Construct Steps

In this section, we use an example to describe how to arrival time according to odes when node comes. To assign numbers to each node to establish a cube structure and construct a hypercube tree in detail.

In this section, we give an example of three-dimensional hypercube to explicit our method and use three figures to gradually make an introduction. Figure 4 illustrates a cube example, assuming that the beginning of the environment has no any node, then we establish a cube according to nodes arrival times. From this example, we can find that 8 nodes can build up a cube and use cube characteristics, each node has the other three sibling nodes to be connected with. Since we presume tree height is 3, Figure 5 illustrates the architecture of 2-level HTS structure. In level 2, we can know that each node not only have three siblings node but also e a parent node, for example, when node 9 comes, it connect with three siblings and one parent: node 10, node 11, node 12 an d node 1.

Besides, each node we assign a number which contains three parts, in Figure 6, X represents the dimension of cube, L represents level of tree. Because of our example is three-dimensional, we can know X=3, L=3 and the number contains 11 bits. Among this number, the front of the two bits represent node at that level, the middle of six bits represent sequence of cube's construction, and the last three bits represent the sequence of nodes. When new node comes, with this number nodes are able to find out three sibling nodes by exclusive OR and find parent node or by shift operation.

After understanding how to construct cube and connection, Figure 7 is our overall system example, as we consider a cube as a super node, we can construct a hyper cube tree, each square represents a cube, finally, we have left tree and right tree including 146 cubes.

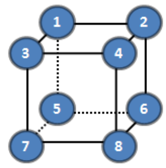

Fig. 4. 1-level HTS structure

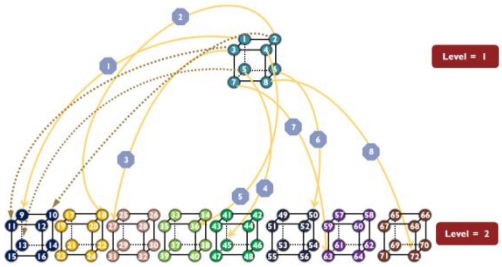

Fig. 5. 2-level HTS structure

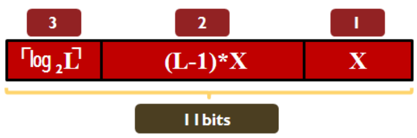

Fig. 6. Number Configuration Diagram

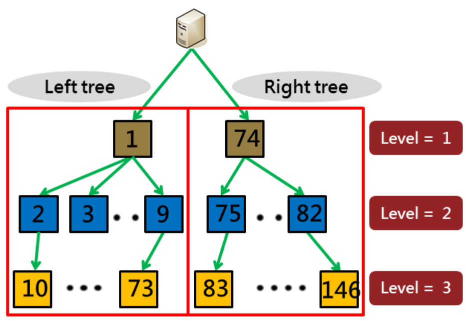

Fig. 7. Overall System

B. Leave processing

In peer-to-peer environment, nodes do leave very frequently, so, in this section we introduce nodes leaving in different level. Due to that we presume tree height is 3, if there is a node wants to leave in level 3, the node should send message to notify its parent node, when parent node receives the message, it returns a response message, after receiving response message , the node can leave. If there has a node wants to leave in level 2, it not only sends message to its parent node but also to its child node which replaces the leaving code. As the same as in level 1, it also sends message to its child node, when child node forwards the received message to the grandchild node, since child node replaces the parent node's function, grandchild node must replacechild node ,too.

IV. Abruptly Recovery Mechanism

In this section, to avoid node's interruption, we propose a timer mechanism to send message to child node and three sibling nodes periodically. If node still in the environment, it will return response message to its parent node, on the contrary, when discover a node interrupt, we can through the backup link to provide the service. The purpose of this is that we hope by the mechanism to solve interruption in peer-to-peer environment.

V. Conclusion and Future Work

In this paper, in order to enhance the tree robustness, we propose a hypercube tree structure for P2P streaming. In HTS, we use backup link to enhance the connections between parent node's siblings and child node's siblings, and use timer mechanism to avoid interruption. In future work, we will design a HTS system that enhances the system robustness and evaluates performance.

References

[1] BitTorrent. [Online]. Available: http://www.bittorrent.com.

[2] PPTV. [Online]. Available: http://www.pptv.com.

[3] PPStream. [Online]. Available: http://www.ppstream.com.

[4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," in *Proc. of ACM SIGCOMM*, Aug. 2002, Pittsburgh, PA, pp. 205-217.

[5] V. N. Padmanabham, H. J. Wang, P. A Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *Proc. of ACM NOSSDAV*, May 2002, pp. 177-186.

[6] D. A Tran, K. A Hua, and T. Do, "ZIGZAG: An efficient peer-to-peer scheme for media streaming," in *Proc. of IEEE INFOCOM*, Mar. 2003, San Francisco, USA, pp. 1283-1292.

[7] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: peer-to-peer patching scheme for VoD service," in *Proceedings of the 12th ACM International World Wide Web Conference (WWW)*, Budapest, Hungary, May 2003, pp. 301-309.

[8] T. T. Do, K. A. Hua, and M. A. Tantaoui, "P2VoD: Providing fault tolerant video-on-demand streaming in peer-to-peer environment," in *Proc. of IEEE ICC'04*, June 2004, vol. 3, pp. 1467-1472.

[9] T. Xu, J. Chen, W. Li, S. Lu, Y. Guo, and M. Hamdi, "Supporting VCR-like operations in derivative tree-based P2P streaming systems," in *Proc. of IEEE ICC'09*, Dresden, Germany, June 2009, pp. 1-5.

[10] X. Zhang, J. Liu, B. Li, and T.S.P. Yum. "Coolstreaming/Donet: A data-driven overlay network for peer-to-peer live media streaming," *In Proc.IEEE INFOCOM*, 2005, vol. 3, pp. 2102-2111.

[11] M. Hefeeda, A. Habib, B. Botev, D. Xu, B. Bhargava, "PROMISE: peer-to-peer media streaming using collectcast," Berkeley, California, USA,*MM'03*,November 2-8, 2003.

[12] V. Pai, "Chainsaw: Eliminating trees from overlay multicast," *IPTPS 2005*, Feb. 2005.

[13] N. Magharei, R. Rejaie, and Y. Guo, "PRIME: peer-to-peer Receiver-driven mesh-based streaming," *IEEE INFOCOM 2007*, May. 2007.

[14] W. P. K. Yiu, X. Jin, and S. H. G. Chan. "VMesh: Distributed segment storage for peer-to-peer interactive video streaming," *IEEE Journal on Selected Areas in Communications*, December 2007, vol. 25, no. 9, pp. 1717-1731.

[15] M. Zhou and J. Liu, "A hybrid overlay network for video-on-demand," in *Proceedings of IEEE International Conference on Communications (ICC)*, Seoul, Korea, May 2005, vol. 2, pp. 1309-1313.

[16] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A hybrid tree/mesh overlay for application-layer live video multicast," *ICDCS 2007*, Sept. 2007.

[17] C. Xu, G. M. Muntean, E. Fallon, and A. Hanley, "A balanced tree-based strategy for unstructured media distribution in P2P networks," in *Proc. of IEEE ICC'08*, June 2008, pp. 1797-1801.

[18] H. Byun and M. Lee, "HyPO: A peer-to-peer based hybrid overlay structure," *IEEE ICACT 2009*, Feb. 2009.

[19] Suphakit, Awiphan, Z. Su and JiroKatto, "ToMo: A two-layer mesh/tree structure for live streaming in P2P overlay network," in *Proc. of IEEE CCNC*, 2010.

[20] N. Huang, Y. Tzang, H. Chang, and C. Ma, "Construction of an efficient ring-tree-based peer-to-peer streaming platform," *NCM2010*, Korea, August 2010.