

Grasping the Flash Crowd in P2P Live Video Streaming Systems

Dr. Pramila Mishra

Department of Applied Mathematics
Barkatullah University, Bhopal (MP), India

Abstract: - This study introduces a distributed algorithm with minimal centralized oversight that arranges newly arrived peers into hierarchical roles to diminish competition between them. This rank hierarchy is subsequently utilized to create various sub-stream trees. The video stream is segmented into sub-streams, with each sub-stream transmitted via a different sub-stream tree. Only the individuals at the highest level of the hierarchy will directly utilize the scarce initial resources and subsequently pass the stream down to those positioned lower in the hierarchy. Consequently, by leveraging the resources offered by the recently arrived colleagues in managing flash crowd, improved system scalability can be accomplished.

Keywords- P2P Network, Content Delivery Systems (CDS), Distributed Hash Map (DHM)

I. INTRODUCTION

The internet has operated on client-server architecture for a considerable period. In these systems, there is a distinct separation between the functions of client and server nodes. The client node initiates the request for the resource (which could be a webpage, music file, video stream, or even a service), while the server node handles these requests and responds with the requested resources. All processing occurs at the server. This architecture functioned effectively as long as the capabilities of client and server nodes varied significantly. However, enhancements in the processing power of local computers and the internet's bandwidth have considerably diminished this gap. The client nodes, fitted with robust processors and quicker internet, transformed from passive receivers to active contributors in the system. Conventional client-server architecture was incapable of harnessing this potential. Furthermore, with the rise in the number of devices linked to the internet, the client-server architecture faced problems related to scalability and single points of failure

1. DHT Overview

A Distributed Hash Table (DHT) based protocol is used to provide the overlay over which the streaming trees can be constructed. In DHT based systems, each resource is identified by a <key,value> pair. The set of all possible keys forms the key space. For e.g., if each key is a m-bit long sequence, the keyspace would comprise of all possible 2^m combinations.

The keyspace is partitioned among the nodes of the network. Each node is responsible for maintaining the resources whose key lies in its region of the keyspace. Finally, a routing algorithm describes how the queries are routed in the network to locate the resources. Different implementations give rise to different systems. Chord [10] is the preferred choice for this thesis but other DHT based protocols such as Tapestry [12] or Pastry [11] can also be used.

2. CHORD

Chord is a scalable peer-to-peer look up protocol designed to construct an overlay network based on DHT. As described earlier, each node as well as resource in the system is identified by a m-bit long node-id and resource-id (key) respectively, with each node maintaining a subset of the resources. By constructing and maintaining efficient routing structures, a resource can be located in a maximum of

$O(\log_2(N))$ hops, where N is the number of nodes in the Chord overlay. The mechanism for mapping resources to nodes and routing queries over the network. On bootstrapping, each node contacts the bootstrapping server and receives a list of existing nodes. It then connects itself in the chord overlay and starts to populate its routing tables by exchanging information with other nodes. Chord supports multiple joins and can bootstrap large-scale DHT networks.

II .CONSTRUCTION OF STREAMING TREES

In this case, the parent Node accepts child n Node as its child and the process is completed for child Node. If the number of fertile children of parent Node becomes k , it sends a deregistration message to $\#(\text{stream}[s] \text{ level}[l-1])$ to remove itself from the feed-forwarder list.

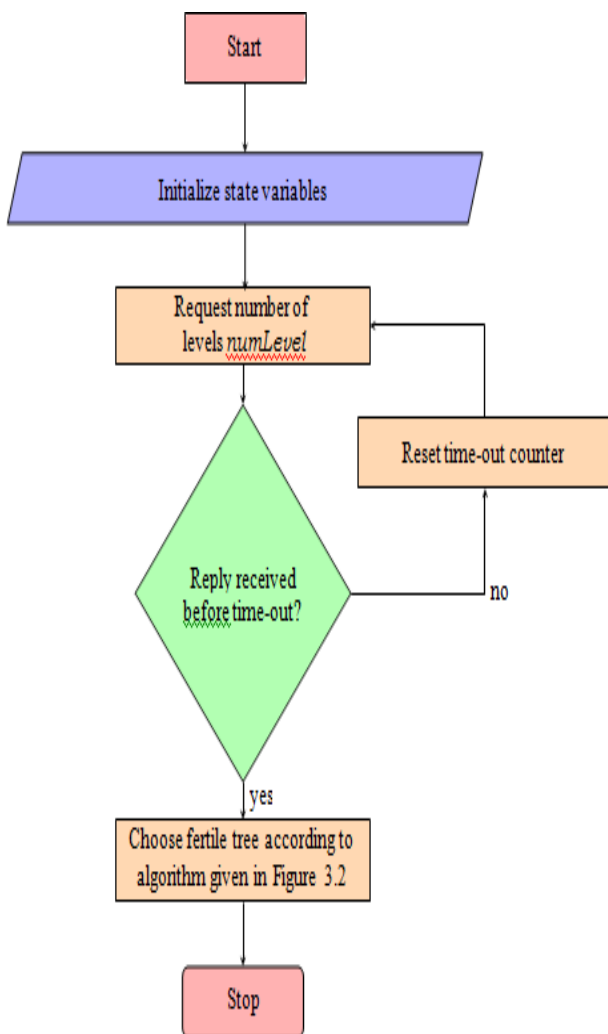


Figure 1 Flow chart for selecting fertile tree.

III. RESULTS

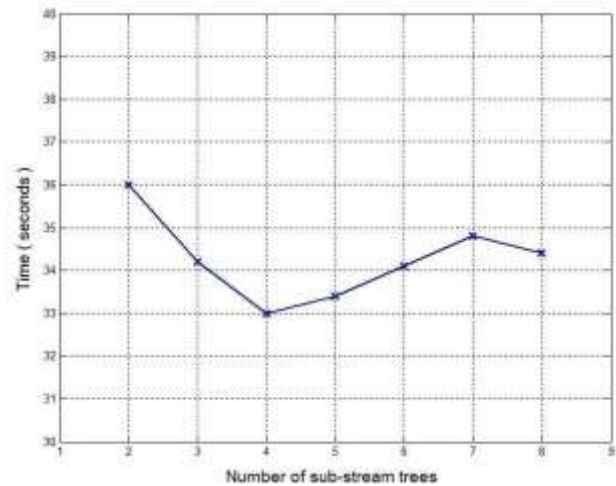


Figure 2 Time taken for 90% of the nodes to get all the sub-streams (stabilization time) for different values of t .

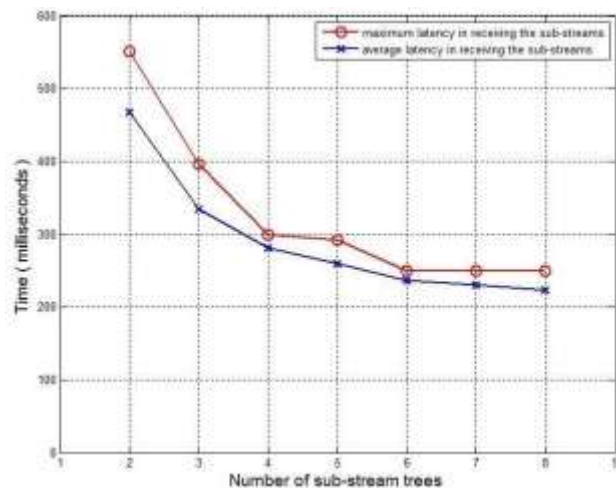


Figure 3 Average and maximum value of average latency at each node for different values of t

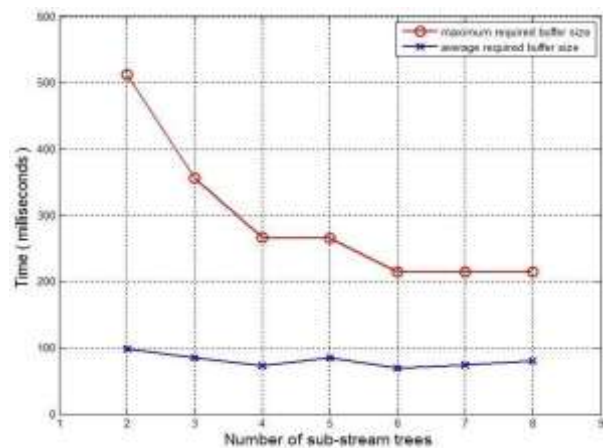


Figure 4 Average and maximum buffer size required at each node.

Author et al. *International Journal of Science, Engineering and Technology*, 2019, 7:3

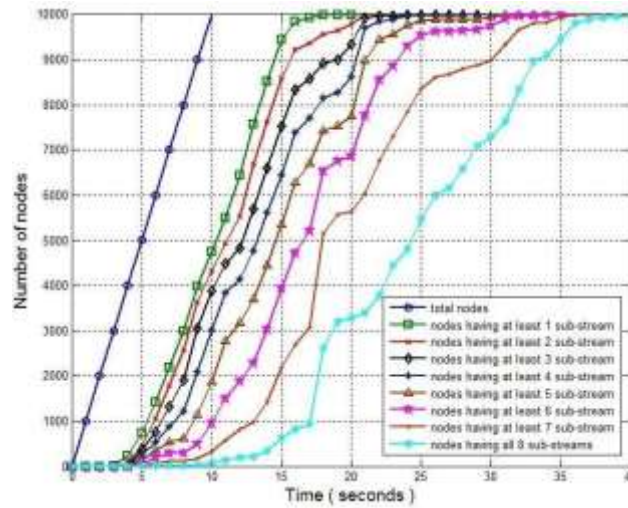


Figure 5 Number of sub-streams received by nodes at different times.

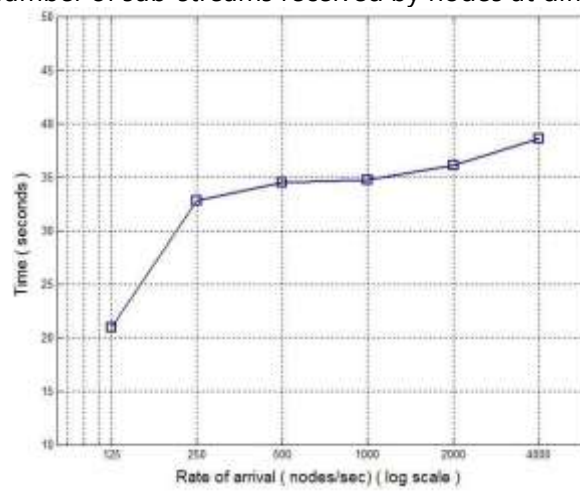


Figure 6 System stabilization time for different node arrival rates.

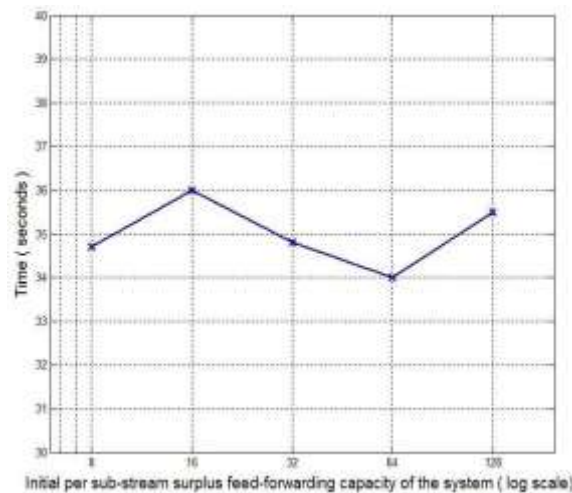


Figure 7 Stabilization time of the system for different values of U_s .

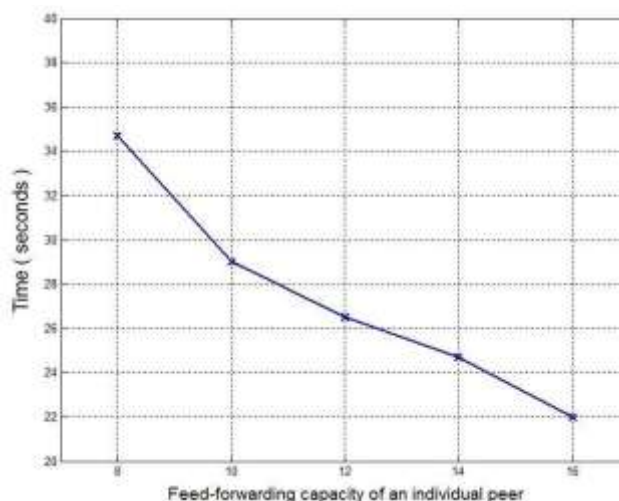


Figure 8 Stabilization time of the system for different values of k.

IV. CONCLUSION

The proposed method consists of two major steps

(1) Arrange the newly arrived peers in different levels of different sub-stream trees. (2) Connect the peers with each other to distribute the stream. The number of levels that are constructed depends upon the node arrival rate. Hence, the design is able to scale to very large number of nodes. From the results, we can see that on increasing the node arrival rate from 2000 nodes/sec to 4000 nodes/sec, the stabilization time of the system increases by only 2.5 seconds.

The slight increase in stabilization time can be attributed to two factor. With higher rate the underlying Chord overlay used for communication takes more time to stabilize. The number of peers in a level increases exponentially. Hence, if the node arrival rate increases, more levels would be created. As a result, number of peers in the lowermost level would be more. This increases competition among the peers belonging to the same level. This can be reduced by creating sub-levels with in levels.

REFERENCES

- [1] R. Schollmeier. —A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications||. In: Peer-to-Peer Computing, 2001. Proceedings. First International Conference on. 2001, pp. 101–102. DOI: 10.1109/ P2P.2001.990434.
- [2] BitTorrent. URL: <http://www.bittorrent.com/>.
- [3] Gnutella.URL: <http://en.wikipedia.org/wiki/Gnutella>.
- [4] Skype — Free Calls to friends and family. URL: <http://skype.com>.
- [5] Xinyan Zhang et al. —CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming||. In: INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies.Proceedings IEEE.Vol. 3.IEEE. 2005, pp. 2102–2111.
- [6] David P Anderson et al. —SETI@ home: an experiment in public-resource computing||. In: Communications of the ACM 45.11 (2002), pp. 56–61.
- [7] Ian Clarke et al. —Freenet: A distributed anonymous information storage and retrieval system||. In: Designing Privacy Enhancing Technologies. Springer. 2001, pp. 46–66.
- [8] Satoshi Nakamoto. —Bitcoin: A peer-to-peer electronic cash system||. In: Consulted1.2012 (2008), p. 28.
- [9] Christos Gkantsidis, Milena Mihail, and Amin Saberi. —Random walks in peer-to-peer networks||. In: INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies. Vol. 1.IEEE. 2004.
- [10] Ion Stoica et al. —Chord: A scalable peer-to-peer lookup service for internet applications||. In: ACM SIGCOMM Computer Communication Review 31.4 (2001), pp. 149–160.
- [11] Antony Rowstron and Peter Druschel. —Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems||. In: Middleware 2001. Springer. 2001, pp. 329–350.
- [12] Ben Y Zhao et al. —Tapestry: A resilient global-scale overlay for service deployment||. In: Selected Areas in Communications, IEEE Journal on 22.1 (2004), pp. 41–53.
- [13] XiaojunHei et al. —A measurement study of a large-scale P2P IPTV system||. In:Multimedia, IEEE Transactions

on 9.8 (2007), pp. 1672–1687.

- [14] Bo Li et al. —Inside the new coolstreaming: Principles, measurements and performance implications||. In: INFOCOM 2008. The 27th Conference on Computer Communications. IEEE. IEEE. 2008.
- [15] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. SScalable application layer multicast.Vol. 32. 4. ACM, 2002.
- [16] Duc A Tran, Kien A Hua, and Tai Do. —Zigzag: An efficient peer-to-peer scheme for media streaming||. In: INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies.Vol. 2.IEEE. 2003, pp. 1283–1292.
- [17] Vivek K Goyal. —Multiple description coding: Compression meets the network||. In:Signal Processing Magazine, IEEE 18.5 (2001), pp. 74–93.
- [18] Miguel Castro et al. —Split Stream: high-bandwidth multicast in cooperative environments||. In: ACM SIGOPS Operating Systems Review. Vol. 37. 5. ACM. 2003, pp. 298–313.
- [19] Dejan Kostić et al. —Bullet: High band width data dissemination using an over- lay ||. In: ACM SIGOPS Operating Systems Review. Vol. 37. 5. ACM. 2003, pp. 282–297.
- [20] Venkata N Padmanabhan, Helen J Wang, and Philip A Chou. —Resilient peer-to- peer streaming||. In: Network Protocols, 2003. Proceedings.11th IEEE International Conference on.IEEE. 2003, pp. 16–27.



Copyright & License:

© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.