International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 4 (2015) pp. 11025-11032
© Research India Publications
http://www.ripublication.com

Load Balancing Mechanism In Content Delivery Networks

Premnath. V
IV year, Dept of Computer Science and Engineering
Sathyabama University
Jeppiaar Nagar, Rajiv Gandhi Road,
Solinganallur, Chennai, Tamil Nadu 600119, India
Email:premnath0909@gmail.com
Mrs.P.Asha
Professor, Dept of Computer Science and Engineering
Sathyabama University
Jeppiaar Nagar, Rajiv Gandhi Road,
Solinganallur, Chennai, Tamil Nadu 600119, India
Email: ashapandian255@gmail.com
Sai Abhishek Kumar
IV year,Dept of Computer Science and Engineering
Sathyabama University
Jeppiaar Nagar, Rajiv Gandhi Road,
Solinganallur, Chennai, Tamil Nadu 600119, India
Email:abhishek.3999kumar@gmail.com

Abstract

In this paper, we propose to derive an effective solution to the problem of Load Balancing in content delivery networks .This system is proposed based on a thorough analysis of a Content Delivery Network system. The result is then used to maximize the performance of the system and also incorporate Time to live (TTL), Piggybacking, and Pre-fetching mechanisms for consistency. Every data item is assigned a corresponding TTL value within which the corresponding data item must reach the destination node. And upon expiration , a call is sent to the data source to revive them. The delay, overhead traffic and consistency ratio are reported versus several variables, where the proposed system is shown to be a cut above the already existing methods. Eventually, the entire system is validated by means of simulations. Our simulation results indicate that compared to a conventional routing protocol, this mechanism can reduce the path discovery overhead , delay and the energy consumed during transmission by a great margin.

Keywords: Content Delivery Networks,Piggybacking,Pre-fetching.

Introduction

A **Content delivery network** (**CDN**)[1-3] is a group of distributed servers (network) that deliver WebPages and other Web related data to a user, based on the topological location of the user by means of a distributed collection of servers. It is an effective approach to support Web Applications and is widely used to minimize congestion issues that occur owing to the immense request rates from the clients , in turn downsizing inactivity and simultaneously augmenting the availability of the content.

- In this proposal, the important issue is the implementation and definition of an efficient solution for the problem of load balancing in “Content Delivery Networks”
- The concept of Piggybacking [4,7], Time to live and Pre-fetching is incorporated to enhance the effectiveness of the proposed system.
- The technique suggested is then validated by means of simulations.

The concept of Piggybacking[5,6] is integrated in the network layer. It is a dual-direction data transportation technique that helps the sender in making sure that the data frame sent by the sender was well received by the intended receiver (ACK acknowledge). This concept of Piggybacking is implemented in this proposal so as to enhance the overall efficiency of the system and to avoid any loss of data packets during transmission.

TTL is a factor that is obtained by the difference in time amongst the query duration of the item and its previous time of updation. The **Time-to-live** (**TTL**) [8] is the amount of hops that a packet is permitted to navigate, prior to being discarded by a router. A packet refers to the fundamental unit of information transport computer networks, and is being implemented in other communications networks as well. And if a particular packet of data is alive in the network for too long without reaching a destination node, that is, if it is subject to an infinite redirection loop, it causes the succeeding packets to wait and thus resulting in an eventual network crash.

Pre-fetching [9] refers to the concept of transferring (data) from main memory to temporary storage so that it is readily available for later use. In this context the temporary storage refers to as the cache. When a sender node intends to send data to the receiver, it initially sends it via intermediate nodes. These intermediate nodes, store the destination node information temporarily and during the next transaction this temporary storage(cache node) provides the destination information to the source, thus facilitating quick processing of data.

Scope:

The significant performance enhancements that we intend to acquire from adapting to such an implementation , are:

1) The overall throughput of the system, that is, the extent to which a message is successfully delivered successfully over a network
2) The time which the clients have to wait in anticipation of a response from the server upon placing a request, is highly reduced.

Drawbacks of Existing System:

- The Local server’s queues tend to overflow due to unawareness of the total number of requests that could possibly be received by the system at a particular instance
- Irregular ordering of distributing the client requests , leading to packet losses at the receivers end.
- There is a possibility that there could be loss of packets due to the overflow of the local server queues.

Problem Statement:

- The Non-Dynamic algorithms choose a server, for service execution, without depending upon the systems status at decision time.
- HTTP redirection allows servers to redirect a client request to a different location .This results in another new trip between the corresponding nodes, causing wastage of time.
- In case of an unconventional traffic scenario[10] characterized by a great quantity of requests, i.e., a flash crowd tends to fill up the system.
- Therefore there is a chance that the system might become unstable, because the rate of input tends to be greater than the rate of service .

New System Proposal:

- The primary step is to model a relevant load-balancing law that promises stability of the queues in a balanced CDN.
- We propose a new mechanism of even distribution of the approaching requests from the clients to the most suitable server, hence stabilizing the total system requests load. This is made possible by frequent interaction among the system nodes.
- Here the client request’s will be gathered into the proxy server and it will distributed into N number of nodes and again it has been uniquely distributed into another proxy.
- We try to avoid the overflow of data in the queue, avoiding the number of redirections mechanisms and increasing the computation speed via a "Relative weight dividend algorithm".
- The concept of Piggybacking and Pre-fetching are included in the proposed system design , so as to make node-to-node interactions even more efficient.

Architecture Diagram

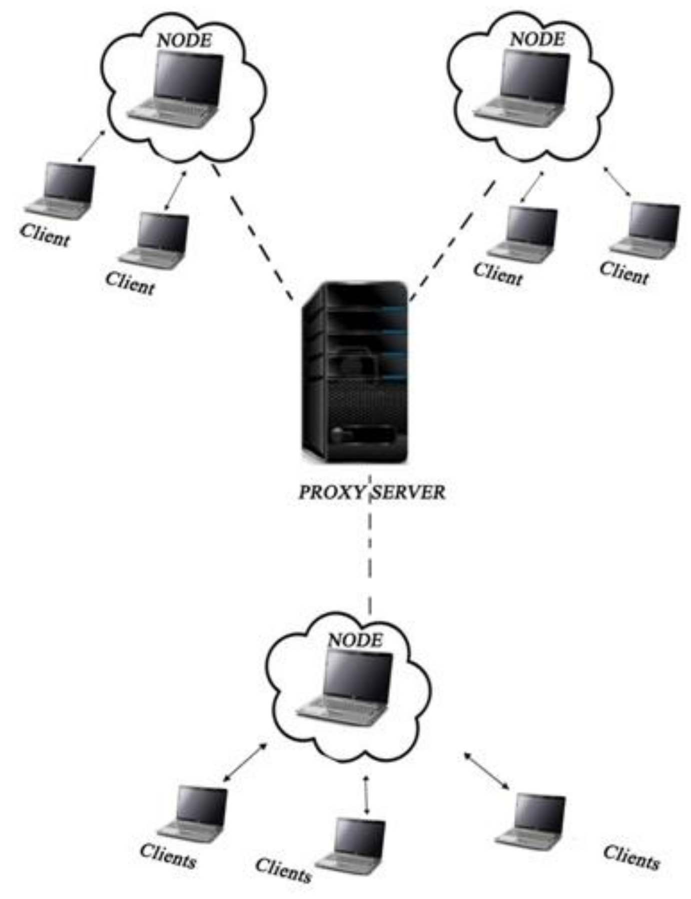

Figure 1: Architecture

Modules

- Node creation
- Implementation of the system
- TTL Adaptation
- Piggybacking
- Pre-fetching
- Client-Server updating
- Performance Analysis

Node Creation

A node (Fig.1) in a network refers to any device in a network that acts as a communication end point. In other words, it refers to any active electronic device, in a network that is capable of transmission of messages in a communication channel .The higher the number of nodes, the higher the number of ways by which the data can be forwarded from the source to the destination.

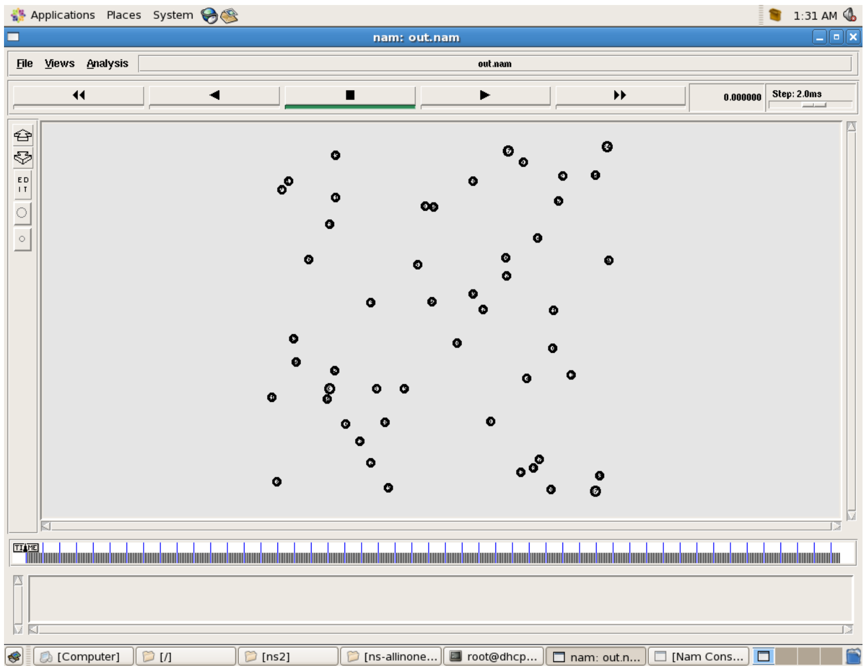

Figure 2: Collection of nodes as seen in a network simulator

TTL Adaptation

TTL is a factor that is obtained by the difference in time amongst the query duration of the item and its previous time of updation. The **Time-to-live** (**TTL**) is the amount of hops that a packet is permitted to navigate, prior to being discarded by a router. A packet refers to the fundamental unit of information transport computer networks, and is being implemented in other communications networks as well. And if a particular packet of data is alive in the network for too long without reaching a destination node, that is, if it is subject to an infinite redirection loop, it causes the succeeding packets to wait and thus resulting in an eventual network crash.

Piggybacking

The concept of Piggybacking is integrated in the network layer. It is a dual-direction data transportation technique that helps the sender in making sure that the data frame sent by the sender was well received by the intended receiver (ACK acknowledge). This concept of Piggybacking is implemented in this proposal so as to enhance the overall efficiency of the system and to avoid any loss of data packets during transmission.

Pre-fetching

Pre-fetching refers to the concept of transferring (data) from main memory to temporary storage so that it is readily available for later use. In this context the temporary storage refers to as the cache. When a sender node intends to send data to the receiver, it initially sends it via intermediate nodes(Fig.3). These intermediate nodes, store the destination node information temporarily and during the next transaction this temporary storage(cache node) provides the destination information to the source, thus facilitating quick processing of data.

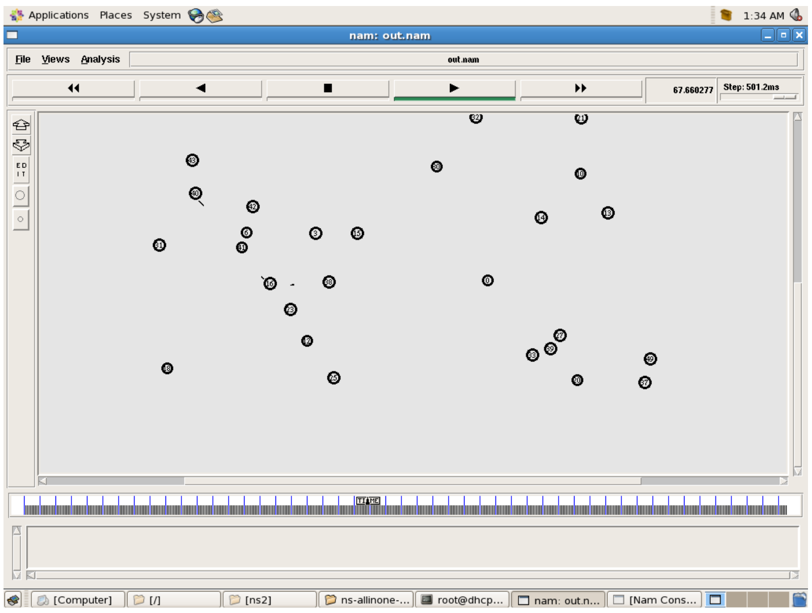

Figure 3: Sending request to cache node as seen in a network simulator

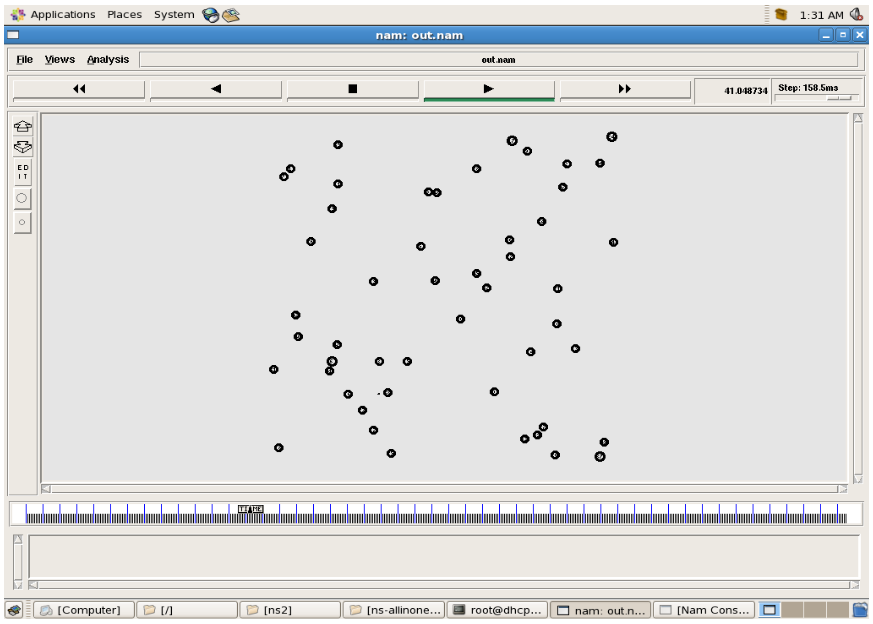

Figure 4: Reply from cache node as seen in a network simulator

Client-Server updating

Upon receiving a message from the receiver, the server checks If the data items have been modified by analyzing their last revised times with those included in the request message sent by the sender .Data Items that have not been modified are considered valid. The data items that have been modified are handled in either of the following ways.

Expired items (those of whose TTL values have expired)and the non expired ones but having the pre-fetch bit set , are revised by sending packets that consist of the actual data items and the related timestamps, to the node where the message originated from . And for the items whose expiry and pre-fetch bits are not set, the server informs them about the missing packets.

Performance Analysis:

The recommended technique is justified by means of simulations and its performance is analyzed by means of graphs(Fig.5), that are plotted based on the behavior of the system .

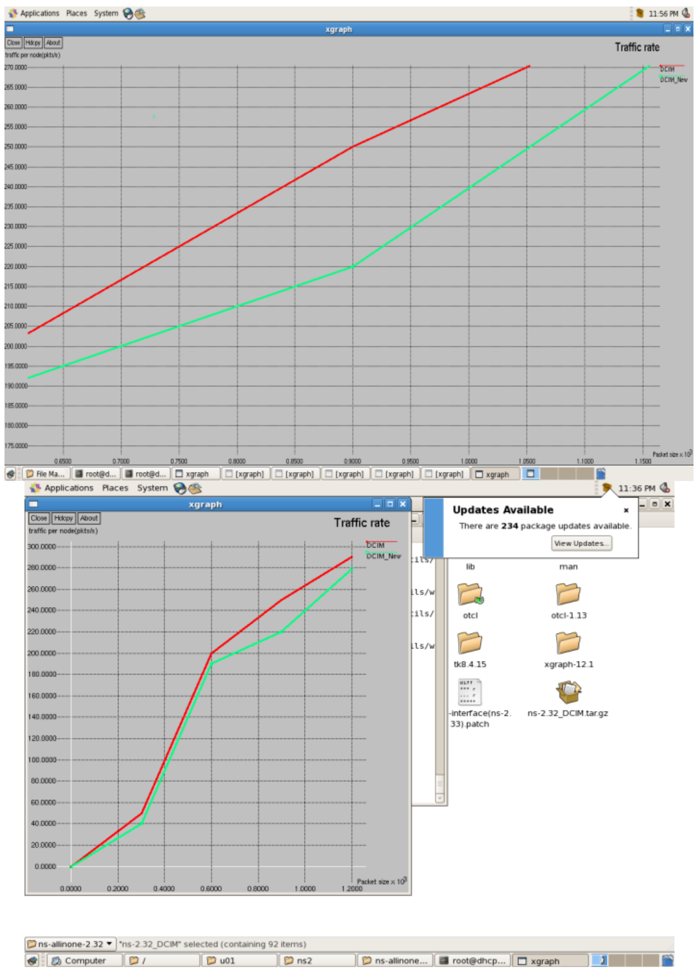

Figure 5: Performance Analysis Graphs

Conclusion

We propose a technique that targets at obtaining load balancing in the network by getting rid of the local queue vulnerability circumstances through redistribution of possible surplus traffic to the set of neighbors of the overcrowded server and also provide ways to retransmit lost data packets back to the intended destination.

References

[1]. Sabato Manfredi, Francesco Oliviero Simon Pietro Romano,"A Distributed Control Law for Load Balancing in Content Delivery Networks" IEEE Netw., vol. 21, no. 1, Feb. 2013.

[2]. Merazka.F ,"Packet loss concealment using piggybacking for speech over IP network services", in Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), 2013 IEEE 7th International Conference , Berlin, Germany,vol.01,pp.509-512.Sep 2013.

[3]. S. Manfredi, F. Oliviero, and S. P. Romano, "Distributed management for load balancing in content delivery networks," in Proc. IEEE GLOBECOM Workshop, Miami, FL, Dec. 2010, pp. 579–583.

[4]. H. Yin, X. Liu, G. Min, and C. Lin, "Content delivery networks: A Bridge between emerging applications and future IP networks," IEEE Netw., vol. 24, no. 4, pp. 52–56, Jul.–Aug. 2010.

[5]. J. D. Pineda and C. P. Salvador, "On using content delivery networks to improve MOG performance," Int. J. Adv. Media Commun., vol. 4,no. 2, pp. 182–201, Mar. 2010.

[6]. M. Colajanni, P. S. Yu, and D. M. Dias, "Analysis of task assignment policies in scalable distributedWeb-server systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 9, no. 6, pp. 585–600, Jun. 1998.

[7]. D. M. Dias, W. Kish, R. Mukherjee, and R. Tewari, "A scalable and highly availableWeb server," in *Proc. IEEE Comput. Conf.*, Feb. 1996, pp. 85–92. 68 IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 21, NO. 1, FEBRUARY 2013

[8]. C. V. Hollot, V. Misra,D. Towsley, andW.Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Trans. Autom. Control*, vol. 47, no. 6, pp. 945–959, Jun. 2002.

[9]. C. V. Hollot, V. Misra, D. Towsley, and W. bo Gong, "A control theoretic analysis of red," in *Proc. IEEE INFOCOM*, 2001, pp. 1510–1519.

[10]. J. Aweya, M. Ouellette, and D. Y. Montuno, "A control theoretic approach to active queue management," *Comput. Netw.*, vol. 36, no. 2–3, pp. 203–235, Jul. 2001.