An AIMD Distributed Control Law for Load Balancing in Content Delivery Networks

Akash B. Rathod
Research Scholar
Government College of Engineering
Aurangabad

Pallavi Kulkarni
Asst. Professor
Government College of Engineering
Aurangabad

ABSTRACT

Content Delivery Networks (CDN) is the best for overcoming the inherited problems face by internet in modern days. The major idea at the basis of this technology is the delivery at edge points of the network, in proximity to the request areas, to advance the user's perceived performance while off-putting the overheads. Literature shows that in Content Delivery Network how this demanding problem of defining and implementing an effective law for load balancing is model. But this system has some drawback, like even if the queue length of server is low they redirect loads to another server to only balance the overall load. Due to this request processing overhead and delay increases. Algorithm model in paper can help to reduce this delay by putting one equilibrium point to request queue. In CDN, the source adjusts its rate using a modified Additive Increase and Multiplicative Decrease (AIMD) algorithm. AIMD has been demonstrated to be sufficient and essential of efficiency and fairness under certain general conditions.

General Terms

Additive Increase and Multiplicative Decrease (AIMD), Content Delivery Network (CDN).

Keywords

Equilibrium point to Server queues, load balancing Algorithm, Additive Increase and Multiplicative Decrease (AIMD).

1. INTRODUCTION

Form last decade large number of user are requesting for wide range of data over internet in the form of text, image, video etc. with respect to this demand there is limited capability of network available to serve, so there is need to use this network efficiently to increase content availability, accessibility and provide congestion control. Content delivery network had provided some solution for the same by adopting a distributed overlay of servers [1].

Content Delivery Network (CDN) represents a popular and useful solution to effectively support emerging Web applications by adopting a distributed overlay of servers. By replicating content on several servers, a CDN is capable to partially solve congestion issues due to high client request rates, thus reducing latency while at the same time increasing content availability. In this literature, in Content Delivery Network the demanding problem of defining and implementing an effectual law for load balancing.

Typical Centralized server architecture and content delivery architecture shown in fig.1. Usually, a CDN (see Fig. 1) consists of an original server (Called *back-end server*) containing new data to be diffused, together with one or more distribution servers, called *surrogate servers*. Periodically, the surrogate servers are actively updated by the back-end server. Surrogate servers are typically used to store static data, while dynamic information (i.e., data that change in time) is just stored in a small number of back-end servers.

The important concern to adopt a Content Delivery Network are 1) increase distributed system throughput, 2) reduce the response time for client request. These two aspects could be contraposition to each other.

The important architecture component of CDN is the request routing mechanism. In request routing process server redirect the client request to appropriate server. It uses the proximity principle by witch request is always redirect to closest server to client. With this principle additionally consider several parameters like traffic load bandwidth and server computational capability.

There are several techniques for request routing depending on the network layers and mechanisms involved in the process of CDN, like DNS request routing ,transport-layer request routing and application-layer request routing[2].

With a DNS-based approach, a specialized DNS server is able to provide a request-balancing mechanism based on well defined policies and metrics [3]–[5]. With transport layer request routing, a layer-4 switch usually inspects information contained in the request header in order to select the most appropriate surrogate server.

With application layer request routing, the task of selecting the surrogate server is typically carried out by a layer-7 application, or by the contacted Web server itself. In particular, in the presence of a Web-server routing mechanism, the server can decide to either serve or redirect a client request to a remote node. Differently from the previous mechanism, which usually needs a centralized element, a web-server routing solution is usually designed in a distributed fashion. *URL rewriting* and *HTTP redirection* are typical solutions based on this approach. In the former case, a contacted server can dynamically change the links of embedded objects in a requested page in order to let them point to other nodes. The latter technique instead exploits the redirection mechanism of the HTTP protocol to appropriately balance the load on several nodes. In this paper, implementation of algorithm will focus attention on the application layer request routing mechanism. More precisely, algorithm will provide a solution for load balancing in the context of the HTTP redirection approaches.

A formal study of a CDN system, carried out through the exploitation of an Additive Increase and Multiplicative Decrease (AIMD) of flow in network of servers[6].

2. SCOPE

In content delivery network generally content are replicated to several sever call surrogate sever, content replicated may be static most of time. CDN involve some orchestrated combination like content delivery, request routing, information spreading and accounting in heterogeneous techniques.

The most important issue in CDN is to achieve load balancing among the servers of network. There are some algorithms proposed in literature [7]. There are different static or dynamic algorithms depending upon their strategy. Static load balancing algorithms are fastest since they don't use selecting process to select servers but in dynamic load balancing strategies uses the information from network and servers to improve assignment process.

The simplest static algorithm is the *Random* balancing mechanism (RAND). In such a policy, the arriving requests are distributed to the servers in the network with a homogeneous probability. Another well-known static solution is the *Round Robin* algorithm (RR). This algorithm selects a different server for each arriving request in a repeated manner.

Each server is loaded with the same number of requests without making any assumption on the state, neither of the network nor of the servers.

The *Least-Loaded* algorithm (LL) is a famous dynamic strategy for load balancing. It assigns the arriving client request to server which at present least loaded. Such an approach is adopted in several commercial solutions. Unfortunately, it tends to rapidly saturate the least loaded server until a new message is propagated [8]. Alternative solutions can rely on *Response Time* to select the server: The request is assigned to the server that shows the fastest response time.

The *Two Random Choices* algorithm [9] (2RC) randomly chooses two servers and assigns the request to the least loaded one between them. A modified version of such an algorithm is the *Next-Neighbor Load Sharing* [9]. Instead of selecting two random servers, this algorithm just randomly selects one server and assigns the request to either that server or its neighbor based on their respective loads (the least loaded server is chosen).

Differently from most of the previous algorithms, a highly dynamic distributed strategy based on the periodical exchange of information about the status of the nodes, in terms of load is propos in literature [10]. By exploiting the multiple redirection mechanism offered by HTTP, Distribute algorithm tries to achieve a global balancing through a local request redistribution process. Any server balances the load locally with its neighbors, and algorithm provides a global balancing strategy by exploiting HTTP multi-redirection process.

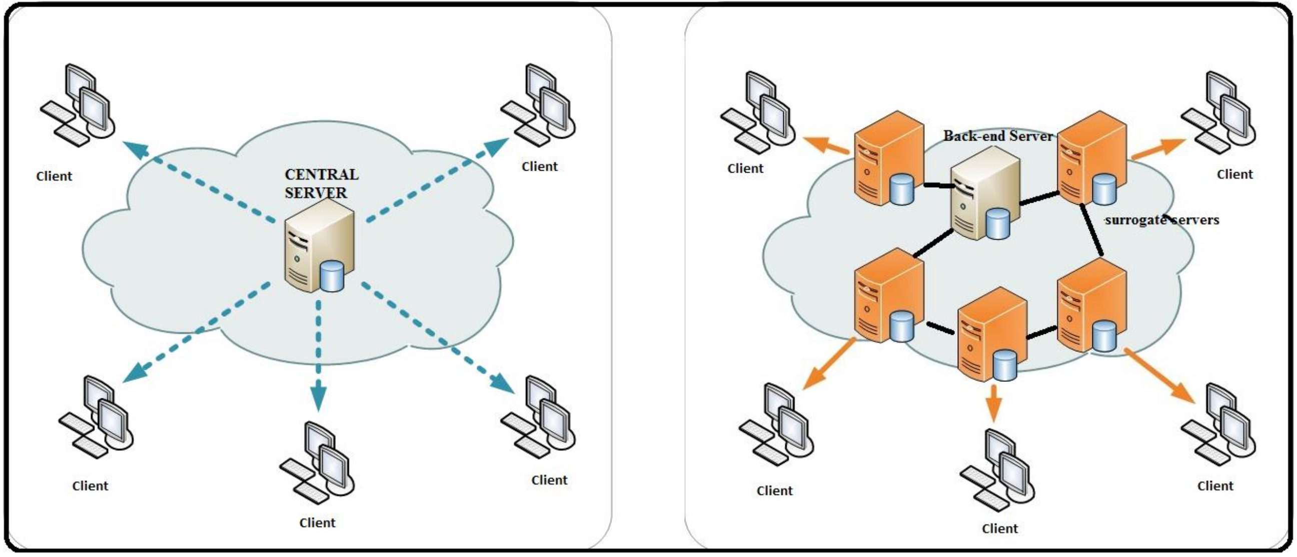

Fig 1: Central Server and Content Deliver Network

3. MODEL FORMULATION: A LOAD BALANCED CDN

CDN infrastructure use to design a novel load balancing law. CDN can be considered as a set of servers each with its own queue. The design of network management law mostly carried out by assuming a continuous fluid flow model of network. This approach is widely use in the communication and control communities (see, for example [11], [12], [13], [14], [15]). This fluid flow model is used for dynamic behavior of each queue. This model is extending to overall CDN system.

Usually, a CDN is designed with adequate resources in order to satisfy the traffic volume generated by end-users. In general, a wise provisioning of resources can ensure that the input rate is always lower than the service rate. In such a case, the system will be capable to efficiently serve all users' requests. Though, work on load balancing in CND focus exclusively on critical conditions where the global resources of the network are close to saturation. This is a realistic assumption since unusual traffic conditions characterized by a high volume of requests, i.e., a flash crowd, can always overfill the available system capacity.

In such a situation, the servers are not all congested. Rather, normally there should have local instability circumstances where the input rate is superior to the service rate. In this case, the balancing algorithm helps prevent a local instability condition by redistributing the excess load to less loaded servers

Let $q_i(t)$ be the queue occupancy of server at time. Consider the instant arrival rate $\alpha_i(t)$ and the instant service rate$\delta_i(t)$. The fluid model (Fig. 3) of CDN servers' queues is given

$$\frac{dq_i(t)}{dt} = \dot{q}_i(t) = \alpha_i(t) - \delta_i(t) \quad \text{….. (1)}$$

Where i = 1…N.

Equation (1) represents the queue dynamics over time. In particular, if the arrival rate is lower than the service rate, there is a decrease in queue length. On the other hand, the queue increases whenever the arrival rate is greater than the service rate. In the latter case, the difference in (1) represents the amount of traffic exceeding the available system's serving rate.

The model described above nicely fits a system in which there is no cooperation among nodes. In such a case, in fact, a node that receives more traffic than it is able to handle will not be able to serve all incoming requests due to an overload condition. It stands clear, though, that such a critical condition might be alleviated if the node in question were allowed to redirect the exceeding incoming traffic to other nodes in the network. Indeed, on the whole system's behavior, situation is paying attention in guaranteeing that the following condition holds:

$$\alpha \leq \delta \quad \text{….. (2)}$$

In the above formula, α and δ represent, respectively, the overall average incoming rate and the overall average service rate of the system once equilibrium is reached. In order to meet the requirement in (2), at same time avoiding local instability situations, there should be able to guarantee that the following condition holds for the entire servers in the network.

Even though the communication protocol used for status information exchange is fundamental for the balancing process, this paper will not focus on it. Indeed, for algorithm mechanism simulation tests, there implemented a specific mechanism: here mechanism extended the HTTP protocol with a new message, called AIMD*CDN*, which is periodically exchanged among neighboring peers to carry information about the current load status of the sending node [16]. Naturally, a common update interval should be adopted to guarantee synchronization among all interacting peers. For this purpose, a number of alternative solutions can be put into place, which is nonetheless out of the scope of the present work.

4. DEFINING ADDITIVE INCREASE AND MULTIPLICATIVE DECREASE (AIMD) FOR CDN

In general load balancing mechanism having two stages, first to update the status of all neighbors load and second will be distributing requests to a less loaded neighbors (servers). Similar system implemented in literature having drawback, like even if the queue length of server is low they redirect loads to another server to only balance the overall load. Due to this request processing overhead and delay increases. Mechanism can minimize this delay by putting one equilibrium point; if queue is reach at this equilibrium point our modified distributed low for load sharing is implemented.

Various possible circumstances are as follows:

If $q_i(t)$ < Equilibrium point: The queue indicates that the queue is going to full. Thus source no needs to perform request redirect mechanism between servers.

• If $q_i(t)$ > Equilibrium point: The large queue indicates that the queue is going to full. The sources are asked to redirect the request from client.

In CDN, the source adjusts its rate using a modified Additive Increase and Multiplicative Decrease (AIMD) algorithm. AIMD has been proven to be sufficient and necessary of efficiency and fairness under certain common conditions.

5. PROPOSED ARCHITECTURE

The algorithm uses a technique like additive increase multiplicative decrease scheme. The novel idea of the algorithm is to update the additive increase term γ and the decrease factor β with respect to the received feedback. We initialize to 1and multiply it by ϕ > 1 each time after a successful transmission. Similarly, we start with β equal to ψ>1 and multiply it by ψ after obtaining a negative feedback. The load balancing algorithm proceeds as follows.

```
AIMD Additive Increase and Multiplicative Decrease
Round 0: /* Initialization. */
ω0 = 1;
β0 = ψ;
γ0 = 1;
Round t:
Send ([ωt]); /* Send window of [wt] packets. */
If Ft = 0 Then /* No loss. */
ωt+1 = ωt + γt;
γt+1 = γ t · ϕ;
βt+1 = βt · ψ ;
Else /* Loss occurred. */
ωt+1 = max (ωt/βt, 1);
βt+1 = βt · γ;
γt+1 = 1;
```

To complete the description of the load Balancing (ϕ, ψ) algorithm it remains to find appropriate ϕ and ψ that will ensure convergence and at the same time allow the algorithm to respond quickly to bandwidth changes. The former can be done by assigning ϕ and ψ values that are close to 1 while the latter is assured by exponential growth of γ and β.

6. CONCLUSION

In this paper, mechanism presented is a novel load-balancing law for cooperative CDN networks. Here networks are defined on model which based on a fluid flow characterization. Mechanism hence moved to the definition of an algorithm that aims at achieving load balancing in the network by overcome local queue instability situation through relocation of probable overload traffic to the set of neighbors of the congested server. The algorithm is first stated in its time-continuous form and then put in a special version intentionally conceived for its real execution and use in a delivery network. Through the help of simulations, here shown both the scalability and the effectiveness of our proposal, which outperforms most of the possible alternatives that have been proposed in the past.

The proposed algorithm attains almost optimal utilization in a steady state providing fairness between competing connections and at the same time responds quickly on bandwidth changes. The present work represents for us a first step toward the realization of a complete solution for load balancing in a cooperative distributed environment.

7. ACKNOWLEDGMENTS

We express our deep thanks to Dr. P. S. Adwani, Principal, Government College of Engineering, Aurangabad. We are thankful to Prof. V. P. Kshirsagar, Head of the Department, computer engineering for his constant encouragement.

8. REFERENCES

[1] H. Yin, X. Liu, G. Min, and C. Lin, "Content Delivery Networks: a Bridge between Emerging Applications and Future IP Networks," *IEEE Network*, vol. 24, no. 4, pp. 52–56, July-August 2010.

[2] A. Barbir, B. Cain, and R. Nair, "Known content network (CN) request routing mechanisms," IETF, RFC 3568 Internet Draft, Jul. 2003 [Online]. Available: http://tools.ietf.org/html/rfc3568.

[3] T. Brisco, "DNS support for load balancing," IETF, RFC 1794 Internet Draft, Apr. 1995 [Online]. Available: http://www.faqs.org/rfcs/rfc1794.html.

[4] 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.

[5] 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.

[6] "Adaptive AIMD Congestion Control" Alex Kesselman School of Computer Science TelAviv University TelAviv, Israel.

[7] S. Meng, L. Liu, and J. Yin, "Scalable and reliable iptv service through collaborative request dispatching," in Proceedings of IEEE International Conference on Web Services – ICWS 2010, July 2010, pp. 179–186.

[8] M. Dahlin, "Interpreting stale load information," IEEE Transactions on Parallel and Distributed Systems, vol. 11, no. 10, pp. 1033–1047, October 2000.

[9] M. D. Mitzenmacher, "The power of two choices in randomized load balancing," IEEE Transactions on Parallel and Distributed Systems,vol. 12, no. 10, pp. 1094–1104, October 2001.

[10] V. Cardellini, E. Casalicchio, M. Colajanni, and P. S. Yu, "The state of the art in locally distributed web-server systems," ACM ComputingSurveys, vol. 34, no. 2, pp. 263–311, June 2002.

[11] C. V. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for aqm routers supporting tcp flows," IEEE Transactions on Automatic Control, vol. 47, no. 6, pp. 945–959, June 2002.

[12] C. V. Hollot, V. Misra, D. Towsley, and W. bo Gong, "A control heoretic analysis of red," in Proceedings of IEEE International Conference onComputer Communications - INFOCOM '01, 2001, pp. 1510–1519.

[13] F. Blanchini, R. L. Cigno, and R. Tempo, "Robust rate control for integrated services packet networks," IEEE/ACM Transactions on Networking, vol. 10, no. 5, pp. 644–652, October 2002.

[14] V. Misra, W. Gong, W. bo Gong, and D. Towsley, "Fluid-based analysis of a network of aqm routers supporting tcp flows with an application to red," in Proceedings of ACM SIGCOMM '00, 2000, pp. 151–160.

[15] D. Cavendish, M. Gerla, and S. Mascolo, "A control theoretical pproach to congestion control in packet networks," IEEE/ACM Transactions on Networking, vol. 12, no. 5, pp. 893–906, October 2004.

[16] Sabato Manfredi, Francesco Oliviero, Simon Pietro Romano, "A Distributed Control Law for Load Balancing in Content Delivery Networks ", Ieee/Acm Transactions On Networking, Vol. 21, No. 1, February 2013 55-68.