International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

Effective Resource Scheduling in Content Delivery Networks Using Dynamic Flow Equilibrium

Resmi .A .M[1], Dr. R. Manicka Chezian[2]

[1]Scholar: Department of Computer Science, NGM College, (Autonomous), Pollachi-642001, India

[2]Associate Professor: Dept. of Computer Science, NGM College (Autonomous), Pollachi-642001, India

Abstract: ***In the current scenario, the technology of the Content Delivery network (CDN) has been extensively applied in all applications. Using CDN technology, a user rapidly gain requested content from the distributed environment. The growth of internet and CDN initiates several performance related issues due to its huge and dynamic contents. Server performance and security based approaches were proposed in CDN, which has the main intension of improving and stabilizing the server performance in the dynamic environment. Even though there are several algorithms and techniques proposed in literature for effective work scheduling in CDN, since the performance has not completed improved. So the system puts forward a new proposal to identify an effective work scheduling technique to improve the quality of the CDN. In this paper, a new work load analysis and task scheduling scheme is introduced, which is named as "Dynamic Flow Equilibrium" a work scheduling scheme in CDN, which handles the dynamic loads and allocates the load to the appropriate content delivery server.***

Keywords: Content Delivery Network, Load balancing, resource allocation, Data Distribution

1. Introduction

Content Delivery Networks are distributed in nature that consist a set of web servers. The main purpose of the CDN is providing effective content with the fastest delivery. The replication is made on CDN to deliver the content faster. Using scenario, a client can access the data, which copied and kept by the CDN or near to the client who accessing the same data from the central server [1]. This process avoids the load of the server. This improves the user experience and reduces the internet traffic around 82%.

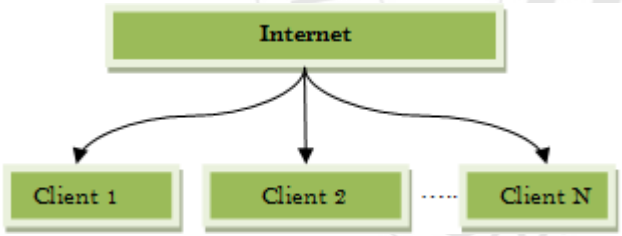

Figure 1: Data delivery process withoutCDN

Data delivery process without using CDN is defined in the figure 1.0. This shows the data delivery made through internet from server to every client. This may increase the load of the server.

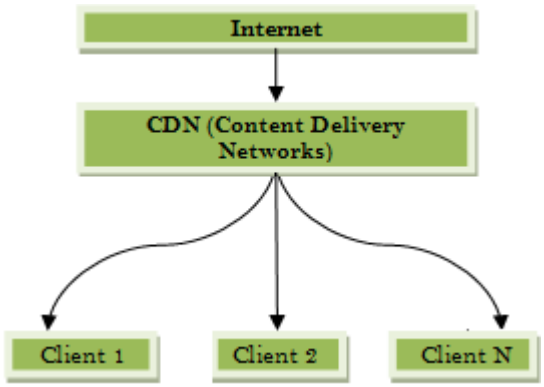

Figure 2: Data delivery process with CDN

Using CDN, the server can deliver data in a faster way. This CDN pre-fetches the data and distributes to the clients. In such case, the CDN find optimal neighbor for fast data collection. This process improves the data delivery ratio.

Server work scheduling is a widely adopted solution to performance and availability problems [2]. Server work scheduling is the process of distributing service requests across a group of servers. End-user requests are sent to a load-balancing device and that determines which server is most capable of processing the request. It then forwards to the request to that server. Server work allocation can also distribute workloads to several firewalls and redirect requests to proxy servers and also caching servers.

Many content-intensive applications have scaled beyond the point where a single server can provide adequate processing powers. And both enterprises and service providers need the flexibility to deploy additional servers quickly and transparently to most end-users. Server work scheduling makes multiple servers appear as a single server – a single virtual service – by transparently distributing user requests among the servers. The highest performance is most achieved when the processing power of servers is used intelligently. Then advanced server load-balancing products can direct end-user service requests to the servers that are least busy and therefore the capable of providing the fastest response times. Necessarily, the load-balancing device must be capable of handling the aggregate traffic of multiple servers. If a server work allocation device becomes bottleneck it is no longer a solution, it is just an additional problem. Another benefit of server work scheduling is its ability to improve application availability. If an applications or server failed, work scheduling can automatically redistribute end-user service requests to other servers within a server farm or to servers in another location [3]. Server work scheduling also prevents the planned outages for software or hardware maintenance from disrupting service to end-users.

Licensed Under Creative Commons Attribution CC BY

Distributed server load-balancing products can also provide disaster recovery services by redirecting service requests to a backup location when a catastrophic failure disables the primary site. End-user requests are sent to a load-balancing device that determines which server is most capable of processing the client request. It then forwards the request to that server. Server work allocation can also distribute to workloads to firewalls and redirect requests to proxy servers and also caching servers. Some applications require persistent sessions between clients and servers. Persistence is the ability to ensure that a user's session with a server will continue to be connected to that particular server. The reasons to preserve a specific session to a particular server can vary from optimizing the cache performance of the server to ensuring a session is not broken. A broken session can result in a shopping cart losing its contents on an e-commerce site [4].

Persistence based on IP destination address enables service providers and web content providers to optimize repetitive web hits to specific content. Persistence based on source IP address ensures that a client remains connected to a specific server for the duration of a state full transaction. Simple persistence based on source IP address works for nearly all Internet applications, except for those clients that might be located behind web proxy farms. It is possible in this scenario for a user's source IP address to change during a single session. This can be overcome using persistence based on the source IP address with a mask. As a result, any sessions from a given set of web proxies will be aggregated to single server.

Work scheduling and load balancing process are very important if CDN has many security considerations, in case of security monitoring, detection and recovery process like EIPDRS developed in [5], need much concentration on performance based issues. In case of security enhancements, performance may need additional concentration.

The efficient work scheduling scheme in distributed system has been developed with the above techniques such as DFE scheme and several workload distribution and queuing techniques. The results of the proposed system have been analyzed in the performance analysis chapter, which differentiates the performance metrics and effective download between several existing approaches. As so this discussed in the above chapters the impact and efficiency of the proposed work scheduling has been described. The time performance analysis considered by several parameters such as throughput, latency, coverage and security. While comparing these with the existing one, the proposed system shows the efficiency by redirecting the client request to the available sub server. The available proxies were filtered by the response time. The server which responds quickly will be considered as a best server and allocates the clients to that system for the load sharing.

In this study, this investigated the performance implications of the Resource allocation scheme with **DFE** in CDN for providing a secure service in a agent-based application server and proposed a Resource allocation scheme for improving server performance through a better resource allocation. The proposed DFE scheme exploits the underlying user-level communication in order to minimize the download time and communication overhead by verifying the resource availability and demand.

2. Related Work

Akamai [6], LimeLight [7],Cloudflare [8] and MaxCDN [9] are eminent and popular CDN providers, which endow with high support to the successful data delivery. The most popular Internet and media companies such as BBC, Microsoft, DreamWorks and Yahoo are the part of the CDN. This also includes many academic such as CoralCDN [10] and CoDeeN [11].

To achieve effective load balancing in distributed servers, request gathering and routing are concentrated. This collects clients request and forwards to the optimal server. To achieve this, several mechanisms have introduced in the literature, such kind of proposals are classified into two categories; one is static and other dynamic. This may depend on the policy adapted techniques. In paper [12] set of policies are defined and adapted for selecting servers. The static technique in server selection doesn't rely on any information about the server status. To achieve the fast decision process, these do not adopt any difficult selection process. This type of technique reduces communication overhead. Butthey are not able to effectively handle flash crowds. Active load balancing strategies represent a valid alternative to static algorithms. This technique makes use of information gathered either from the network or from the servers in order to improve the request assignment process.

The selection of the appropriate server is done through a collection and subsequent analysis of several parameters extracted from the network elements. Consequently a data exchange process among the servers is needed, which unavoidably incurs in a communication overhead.

The most conventional load balancing algorithms used in CDN are RAND (Random balancing mechanism), which is a static algorithm and relies on set of policies. The client requests are uniformly distributed using this technique. The one more static algorithm is RR (Round Robin), this will select different server at each request in a cyclic way. This allocates work load equally for the entire server. RR doesn't consider the current state of the server before allocation.

While considering a dynamic algorithm, Least-Loaded algorithm (LL) is a popular one, which follows dynamic strategy for load balancing. This collects client requests and assigns to the least loaded server. This type of solution is commonly used every solution. But this solution needs to send more requests rapidly to the least loaded server until they get response [13]. Some authors [14] used response time based server selection.

The author in [15] proposed a new random choice algorithm named as 2RC, two random choices algorithm. This chooses two least loaded servers and allocates to any one among the two. This is ineffective and increased overhead, so an extension of the 2RC is developed in [16], a Next-neighbor load sharing. This selects a server randomly and allocates the client to the server based on their load.

Licensed Under Creative Commons Attribution CC BY

An alternative solution for load balancing and appropriate server selection in CDN, which is a dynamic work scheduling mechanism. A highly dynamic distributed strategy based on the response time with the consideration of information about the status of the server. The loadcalculated in terms of number of requests, task, and task completion time and resource availability. By developing the parameter based load allocation mechanism, this algorithm tries to achieve a global balancing through different types of agents. Upon arrival of a new request, indeed, a CDN server can either elaborate locally the request or redirect it to other servers according to the response time of each server, which is based on the response, server and its state information'scollected by the different agents.

3. Proposed System

High and low loaded servers and balance the load among others. The distributed dynamic work scheduling system is a set of surrogate physical and virtual machines to balance the work load. The proposed technique gives an effective way to overcome the work scheduling problem in CDN. So the system introduced a new dynamic work scheduling scheme named as "**Dynamic Flow Equilibrium**" a work scheduling scheme in CDN, which handles the dynamic loads and allocates the load to the appropriate content distribution server. This has been performed by receiving the customized QOS requirements of the user. The QOS attributes such as time, cost, delay, security etc. When a client *request is* made, the LMS (load monitoring server) and LRS (Load report Server) will perform workload identification and reports to the client, based on the current and future load, the system transmit the load to the optimal server. LMS and LRS uses bloom search algorithm to reduce the resource management cost. DFE presents a new mechanism for redirecting incoming client requests to the most appropriate server based on user customized parameters, thus balancing the overall system requests load. The mechanism leverages local balancing in order to achieve global balancing. This is carried out through a periodic interaction among the system nodes.

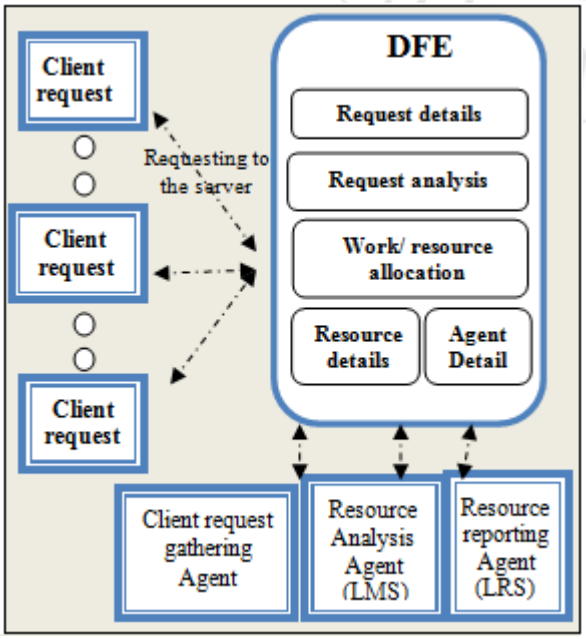

Figure 3: Load balancing architecture in CDN

The DFE model is aimed at mitigating the limitation of existing technique by using dynamic mechanism to achieve work load management and improving CDN performance. This system deploys a dynamic Resource analysis in the distributor to obtain the load information from each application server. The distributor updates the resource information every 300 ms. since the cluster uses a low-overhead user level communication and the communicating messages related to the load information are relatively short, the communication overhead is not significant. The server load will be calculated by determining some elements such as number of connections made in the server, server allocation. The approximate server weight consist with the number of open connections, here the concept cluster has been proposed to combine all requesting clients for a particular server at a time by the agents.

The server capacity should mention with the system to identify the exact load or maximum load to the server at the server implementation.

The load of the i^{th}server L_i is calculated by the number of open connections. If a cluster consists of N servers, where the i^{th} node is denoted by n_i, then L_i = wd _ rd , Here, wd and ws are weight variables for dynamic requests and static requests, and ws-rst $_i$, rd_i and rst_i are the number of outstanding dynamic and static requests of n_i, respectively.
wd=dynamic requests
ws=static requests
rd=Number of outstanding dynamic request
rs=Number of outstanding static request
L_i=Number open connection
rst_i= the number of outstanding dynamic and static requests of n_i,
C =Open connection
$L_i=C_1+ C_2+ C_3…….C_n$selected server from K_N
Update L_i = wd–rd (weight variables of dynamic request-number of outstanding dynamic requests)
Ln_i=(rs - rd)

These weight variables can be adjusted. This assigns the average processing time of the static and dynamic requests to the weighted values for each request to calculate the load of the i^{th} server and thresholds T_1 and T_2. To forward requests, servers use two thresholds T_1 and T_2. If $L_i > T_1$ then n_i forwards requests along with the resource details to one of the servers whose load is less than T_2.
Average processing Time=A
Session key S_k
Selected Server $s=K_{ni}(T_1,T_2…T_n)$
Kni= average load of server s_i
If Li < T2 for every (S_k)assignS

In general, the quality of resource allocation is statistically determined by the number of client's requests. As the statistical nature of the client population fluctuates, the evenness of resource allocation can be observed to vary slightly over time. It is important to note that achieving precisely identical resource allocation on each cluster host imposes a performance penalty (throughput and response time) due to the overhead required to measure and react to load changes. This performance penalty must be weighed against the benefit of maximizing the use of cluster

Licensed Under Creative Commons Attribution CC BY

resources. In any case, excess cluster resources must be maintained to absorb the client load in case of failover. Server Resource management takes the approach of using a very simple but powerful dynamic algorithm that delivers the highest possible performance and availability.

4. Results Analysis

From the literature and its results it can observe that existing RR shows the worst performance since subsequent requests from a client are not likely to be forwarded to the same server that caches the previous session information of the client. Thus, CPU cycles are wasted to re-authenticate and negotiate keys between a client and a server. The results of RR show that the CDN setup procedure is the main bottleneck in application servers.

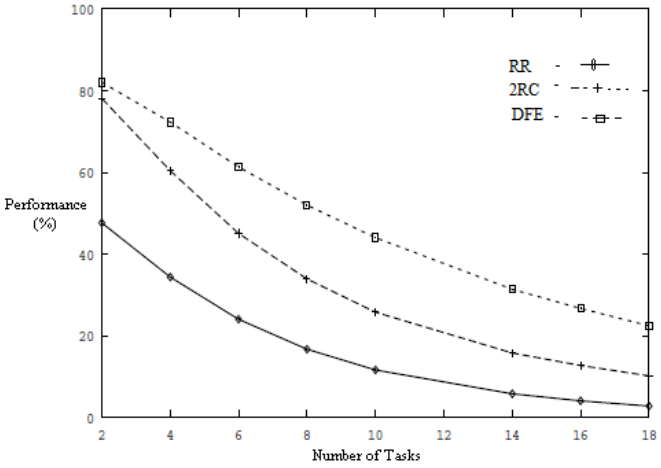

Figure 4: Performance comparison between load balancing algorithms in terms of tasks

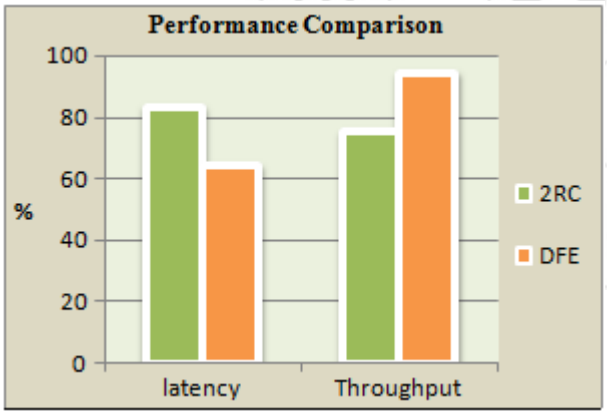

Figure 5: Performance comparison chart

Therefore, this experiment indicates that, in a cluster based Web server, which provides CDN connections, good performance cannot be obtained with a distribution policy that considers only the cache locality or resource utilization. User locality for the reuse of the session information is the critical factor to improve performance. Figure 4.0 also shows that the proposed Dynamic Flow Equilibrium scheme has improved performance than the existing algorithm 2RC and RR.

Although the difference is not clear in the Figure5.0 because the latency of the RR distributor is too high compared to the other two models, the latency of Dynamic Flow Equilibrium is about 50 percent less than that of the 2RC .Like the latency result, the throughput of RR is much lower compared to the Dynamic Flow Equilibrium and 2RC models. The Dynamic Flow Equilibrium model also yields a better throughput compared to FE based on the load increases. The following table shows the Load metrics of Dynamic Flow Equilibrium algorithm.

Table 1: DFE Load Metrics

Range Number	Load Numbers	Step Size (ms)	Maximum Response Time (ms)	Maximum Response Time (h:m:s)
1	2-15	2	32	0: 0: 0
2	16-31	4	96	0: 0: 0
3	32-47	8	224	0: 0: 0
4	48-63	16	480	0: 0: 0
5	64-79	32	992	0: 0: 0
6	80-95	64	2016	0: 0: 2
7	96-111	128	4064	0: 0: 4
8	112-127	256	8160	0: 0: 8
9	128-143	512	16,352	0: 0:16
10	144-159	1024	32,736	0: 0:32
11	160-175	2048	65,504	0: 1: 5
12	176-191	4096	131,040	0: 2:11
13	192-207	8192	262,112	0: 4:22
14	208-223	16,384	524,256	0: 8:44
15	224-239	32,768	1,048,544	0:17:28
16	240-254	65,536	2,031,584	0:33:51

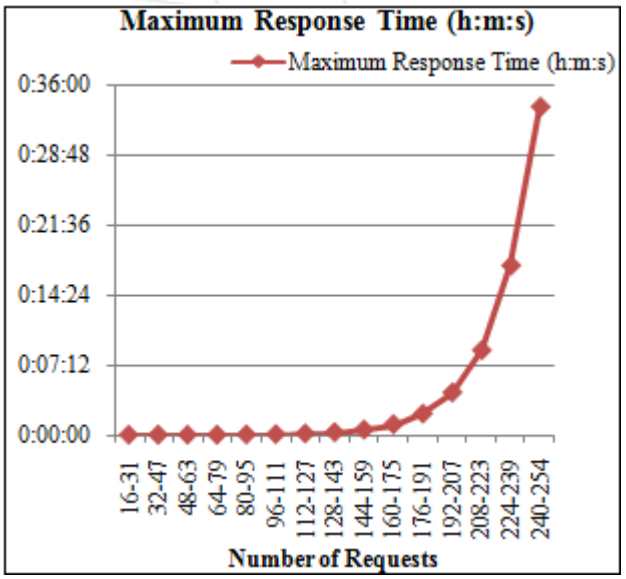

Figure 6: Response Time for Load in numbers

The Figure6.0 represents maximum response time according to the load of the server. When the server load exceeds its limit automatically the server performance degraded. To overcome this issue the proposed system shows the server allocation according to the response time.

5. Conclusion

Performance monitoring and service enhancement in CDN is an inevitable process for lucrative CDN services. Web and content provider's aims to achieve the quality of the services with minimum cost and complexity, But server replication and contributing more server resources are less preferred by

Licensed Under Creative Commons Attribution CC BY

the CDN providers. This paper developed a new server work balancing and process allocation system for fast and uninterrupted, high QOS CDN services. Using the simple methodology on resource selection and allocation, the CDN service provider can improve their services. This approach gives the optimal resource selection and fast content delivery process in the high traffic channels at the high demand epoch. Even though several work scheduling and resource selection techniques were introduced, the entire process needs pre-analysis and post- revising process. This increases the communication overhead and complications.

References

[1] 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.
[2] 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.
[3] 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.
[4] Z. Zeng and B. Veeravalli, "Design and performance evaluation of queue-and-rate-adjustment dynamic load balancing policies for distributed networks," IEEE Trans. Comput., vol. 55, no. 11, pp. 1410–1422, Nov. 2006.
[5] Resmi, A. M., and R. Manicka Chezian. "An extension of intrusion prevention, detection and response system for secure content delivery networks." *Advances in Computer Applications (ICACA), IEEE International Conference on*. IEEE, 2016.
[6] Akamai, "Akamai," 2011 [Online]. Available: http://www.akamai. com/index.html
[7] Limelight Networks, "Limelight Networks," 2011 [Online]. Available: http://.uk.llnw.com
[8] cloudflare, "cloudflare," 2017 [Online]. Available: http:// https://www.cloudflare.com/
[9] maxcdn, "maxcdn," [Online]. Available: http:// https://www.maxcdn.com/
[10] Coral, "The Coral Content Distribution Network," 2004 [Online]. Available: http://www.coralcdn.org
[11] Network Systems Group, "Projects," Princeton University, Princeton, NJ, 2008 [Online]. Available: http://nsg.cs.princeton.edu/projects
[12] V. Cardellini, E. Casalicchio, M. Colajanni, and P. S. Yu, "The state of the art in locally distributed Web-server systems," Comput. Surveys, vol. 34, no. 2, pp. 263–311, Jun. 2002.
[13] M. Dahlin, "Interpreting stale load information," IEEE Trans. Parallel Distrib. Syst., vol. 11, no. 10, pp. 1033–1047, Oct. 2000.
[14] R. L. Carter and M. E. Crovella, "Server selection using dynamic path characterization in wide-area networks," in Proc. IEEE INFOCOM, Apr. 1997, vol. 3, pp. 1014–1021.
[15] M. D. Mitzenmacher, "The power of two choices in randomized load balancing," IEEE Trans. Parallel Distrib. Syst., vol. 12, no. 10, pp. 1094–1104, Oct. 2001.
[16] C.-M. Chen, Y. Ling, M. Pang, W. Chen, S. Cai, Y. Suwa, and O. Altintas, "Scalable request routing with next-neighbor load sharing in multi-server environments," in Proc. IEEE Int. Conf. Adv. Inf. Netw. Appl., Mar. 2005, vol. 1, pp. 441–446.

Author Profile

*Resmi.A.M receivedMCA from Madurai Kamaraj University, Madurai. She completed her M.Phil Degree from Bharathiar University, Coimbatore. Currently she is doing Ph.D in Computer Science at NGM College, Pollachi. India. She has 10 years of Teaching Experience. She has published 5 papers in national level/international conference and journals. Her research interest includes in the areas of Advanced Computer Network, Data Mining and Image Processing.

Dr. R. Manickachezian received his M.Sc., degree in Applied Science from P.S.G College of Technology, Coimbatore, India in 1987. He completed his M.S. degree in Software Systems from Birla Institute of Technology and Science, Pilani, Rajasthan, India and Ph.D degree in Computer Science from School of Computer Science and Engineering, Bharathiar University, Coimbatore, India. He served as a Faculty of Maths and Computer Applications at P.S.G College of Technology, Coimbatore from 1987 to 1989. Presently, he has been working as an Associate Professor of Computer Science in N G M College (Autonomous), Pollachi under Bharathiar University, Coimbatore, India since 1989. He has published one-fifty papers in international/national journal and conferences: He is a recipient of many awards like Desha Mithra Award and Best Paper Award. Recently he received the award "Best Computer Science Faculty of the Year 2015" from Association of Scientists, Developers and Faculties. His research focuses on Network Databases, Data Mining, Distributed Computing, Data Compression, Mobile Computing, Real Time Systems and Bio-Informatics.

Licensed Under Creative Commons Attribution CC BY