Flow Count: A CDN Dynamic Replica Placement Algorithm for Cross Traffic Optimization

Moises Rodrigues, Andre Moreira, Marcio Neves, Ernani Azevêdo, Djamel Sadok
Network and Telecommunications Research Group - GPRT
Recife, Pernambuco, Brazil
Email: {moises,andre,mmn2,ernani, djamel}@gprt.ufpe.br

Arthur Callado
Quixadá Campus - UFC
Quixadá, Brazil
Email: arthur@ufc.br

Victor Souza
Ericsson Research
Kista, Stockholm, Sweden
Email: victor.souza@ericsson.com

***Abstract*—Content Distribution Networks (CDN) are a popular technology to deliver content and have attracted great interest in recent years. Replica Placement Algorithms (RPA) are one of the main widely studied CDN strategies. In this paper we propose a new dynamic RPA strategy, very similar to the Greedy strategy, based on the count of data flows through network nodes. Our experiments show better results using the proposed Flow Count Strategy than using Greedy or Hotspot algorithms when considering cross traffic. Also, the obtained results show that Flow Count seems to place replica servers more efficiently during local flash crowd events.**

Keywords-component; Content Distribution Networks; Replica Placement Algorithms; Discrete Event

I. INTRODUCTION

The Internet plays a key role in our modern society. Its usage is increasing every day resulting in new and renovated challenges. Using CDNs [1], accessibility has improved through content replication in replica servers near clients [2]. The success of CDNs is reflected by the amount of Internet traffic companies such as Akamai claim to handle reaching between 15 and 20% of the world's Web traffic.

One of the important CDN research problems consists in deciding where to place content. The techniques applied to this problem fall in two sets: caching algorithms and replica placement algorithms (RPA). The former class consists of distributed algorithms that perform content placement within replica servers and is also known as caching replacement techniques or simply caching technique [1]. The latter class focuses on choosing the best location of replica servers, thus, reducing the perceived latency and bandwidth consumption. This work is concerned with the second class of problems, namely, dynamic replica placement.

The CDN can be modeled using a number of nodes, replica servers, clients and a routing matrix. The output of RPA is a placement matrix linking the replica servers to the nodes, representing the optimal location(s). The replica placement problem consists basically of a cost function that has to be optimized under certain constraints (number of replicas, server capacity or client quality of experience). The cost is calculated according to a certain placement matrix in terms of user perceived latency, traffic generated or similar parameters. The problem of replica servers placement is NP-complete [7]. RPAs can be divided into two categories, static and dynamic algorithms. Unlike static RPAs where replica server location is fixed, dynamic strategies adapt such locations according to scenario changes.

The main contribution of this paper is to propose a new dynamic strategy for placing replica servers over a network that takes into account the number of data flows through every node. Our experiments show that using the Flow Count strategy, we can reduce cross traffic when comparing to other well-known approaches from the literature.

The remainder of this paper is organized as follows. Related work of RPA is discussed in Section 2. Section 3 describes the simulation tool used in our experiments. Section 4 describes the Flow Count strategy followed by simulation results in Section 5 and Section 6 with lessons learned and finally conclusions in Section 7.

II. STATE OF THE ART

Some theoretical approaches model the RPA problem as the "center placement problem": for the placement of a given number of centers, they minimize the maximum distance between a node and the nearest center. Some variants of this problem are the facility location problem, k-hierarchically well-separated trees and the minimum K-center problem [12]. The minimum k-center problem is highly complex and computationally intensive to be used in practice. Due to the computational cost of these algorithms, some heuristics have been proposed. They take into account the existing information from a CDN, such as workload patterns and the network topology, providing sufficient solutions with a lower computation cost. The work in [13] evaluated some of them. These were characterized along three axes: metric scope (the technique used like centralized or decentralized computation), approximation method (e.g. ranking, relaxation, fixed threshold, dynamic programming, etc) and cost function simplification.

During the last decade, there has been a considerable amount of research papers on replica placement. The first algorithms fit in the group of static placement [8][9][10]. The best representatives are the Greedy [10] and HotSpot [8]. The Greedy works by placing M replicas among N potential sites, one replica at a time. In each iteration, each of N potential sites is evaluated in order to determine the suitability of a site to host a replica. The iteration continues until M servers have been chosen. The Hotspot strategy simplifies the process by choosing the sites that generate the greatest load according to a degree of vicinity.

The next group is the dynamic replica placement [5][6][13]. It is an improvement of the static approaches since they suffer from a lack of adaption to changes in user requests. The work in [11] discussed a robust replica placement for improved performance under the uncertainty of random server failures while [14] it also introduced QoS awareness. In [7], a model to reduce the computational cost of the heuristics is presented for problems with limited storage capacity constraints.

III. SIMULATION TOOL

The present Simulator, the P2PCDNSim [19], adopts a number of abstraction levels enabling its modular extension. Right from the start, we made an important decision in the design of our CDN simulator. We knew beforehand that there are many ways in which a CDN may be built. Some are based on fixed infrastructure, others use P2P, are hybrid or Cloud based. As a result, there is clear separation between these different overlay "operational modes" or paradigms and the common underlying support strategies and functionalities which include caching strategies, placement algorithms, the simulator's event scheduler, metric collection functions and the plotting of graphs. It turned out that this was a good choice as we could expand our scenarios to some initially unforeseen ones such as the study of CDNs with and without virtualization support, Hybrid CDN-P2P systems and multiple cooperating CDNs within a single topology.

A second adopted design decision was the support given to geographical location through an advanced realistic globe shaped interface. A user may examine in real-time relevant metrics for example by monitoring their color coded evolution on the graphs. Animated output conveys important knowledge and allows the accompaniment of the observed system's evolution at a controlled pace. The reader is invited to test our simulator at cdn1.gprt.ufpe.br through some guided scenarios.

To speed up development, the Desmo-J (2011) framework and JFreeChart (2011) were used for modeling discrete events and drawing graphs. From a structural perspective, P2PCDNSim has been divided into three main parts: core, network and overlay. At its core, we find the clock, an event scheduler and data collectors. Above this lies the network level that mimics a communication network, topology configuration, protocols and links properties such as their symmetry, capacity, delay, packet loss, congestion, jitter and throughput information. The third abstraction level is that of the overlay CDN, i.e., the actual object of a simulation. Processes make up the entities at the first layer which are then mapped onto their discrete events. The main drawback is high memory usage due to its level of faithful portraying of functionalities such as the underlying communication protocols and real-time animation. For example, to each packet, a Java object is associated with all the inherent overhead. A router's queue maintains these in memory. Nonetheless, low memory costs should mitigate this problem in the future.

A. CDN Architecture and Replica Placement Concept

A CDN is a highly complex overlay network composed by a set of elements that act in a coordinated way to achieve efficiency and provide QoE to the end-user. Application-specific servers and caches at several places in the network handle the distribution of specific content types (e.g., Web content, streaming media, news information, and real time video) [1].

In a virtualized model, the replica servers may be virtual machines instantiated in a Cloud computing environment, for example. Thus, new replicas can be created and added to the CDN infrastructure as operating conditions change. But, most importantly, replicas can be repositioned, adapting the infrastructure to new user demands and resource usage states. Thus, for example, during a flash crowd event it should be possible to replace surrogates to keep them closer to the source of the event, decreasing the startup delay experienced by clients. This replacement is made based on several metrics and information and is done during the simulation within a time window T. Thus every time window the replacement algorithm is executed and according to new values collected for the important metrics replicas are replaced. It is important to notice that after the replacement old clients still receive information from the old server but all new clients are redirected to new replica servers.

IV. ALGORITHM DESCRIPTION AND ANALYSIS

In this section we present more detailed description of the Flow Count strategy, divided in two subsections. The first subsection presents the strategy in a more detailed fashion describing the reallocation strategy functioning and showing an example of how we count flows. The second subsection shows the complexity analysis of the algorithm.

A. Flow Count Strategy

Our flow count placement strategy is based on a greedy algorithm with a specific selection function. This selection function uses flow counts as its main metric to decide upon the placement. Considering a data flow in a CDN the perfect scenario is when there is only one flow for each object between the origin server and each replica server. The Flow Count placement strategy follows this idea and counts all flows passing through all routing nodes in the topology then later uses this information to place replica servers to minimize overall flow count.

The strategy is divided in two basic parts. The first one is counting flows and the second one is analyzing the flow counts to decide where to place servers. The first part is made by an analyzer running in every routing node that counts and updates tables with information about all flows passing through that node. A flow is represented by the object identifier and the destination address. This information is stored to prevent counting again an already recognized flow. Recent works involving both Content Aware Networks (CAN) [18] and new network management tools [17] presents new horizons that made collecting this information possible.

The second part of the strategy decides where to place servers based on two different pieces of information: topology and flow counts. The topology represents the network and states where servers can be placed. Flow counts show where traffic is crucial, it reflects the importance of some network nodes for the overall content distribution. It is important to notice that all counts are cleared after every replacement made. The pseudo-code in Figure 1 illustrates the placement selection process for Flow Count placement strategy. The process starts with the collection of all flow counts. At every T seconds, for each candidate node, the algorithm calculates the cost for server requests considering all placements already made, if there were any, and the new candidate placement. This cost is calculated as the product of the count of flows on the candidate node with the distance to every other node. Overall configurations costs are ordered and replicas are placed according to the lower costs configuration.

Algorithm 1: Allocation of virtual nodes

```
Inputs: nodeList, flowCountsList, placedReplicasList;
Output: bestNode;
begin
for each candidateNode nodeList do
    for each node nodeList do
        flowCount = flowCountsList ( node );

        for each replica placedReplicasList do
            replicaCost = flowCount * distanceTo(replica);
            if replicaCost < costToClosestReplica then
                costToClosestReplica = replicaCost
        costToCandidate = flowCount * distanceTo(candidateNode);
        cost <- min(costToClosestReplica, costToCandidate);
    if cost <lowerCost then
        bestNode<- node;
        lowerCost<- cost;
    end
end
return bestNode;
```

Figure 1. Algorithm for selection of the best node to place a replica server.

V. FLOW COUNT SIMULATION RESULTS

This section describes simulation results for the Flow Count's strategy using P2PCDNSim. First we describe the scenarios and then show results for each scenario used.

A. *Simulation Scenarios*

For our results we used a scenario, called ProwGen scenario, generated using ProwGen [4]. Our experiments run on top of the 500 nodes topology generated using BRITE [3] topology generator divided in 10 ASes. Also, in the ProwGen scenario clients had 1MB/s links while routers and other CDN entities had 1GB/s links. It has almost a million requests and two flash crowds during the timeline. TABLE I illustrates this scenario.

B. *Metrics*

During our simulations we collect a series of metrics to evaluate and compare the strategies used. This section describes all the collected metrics and what they represent.

Total Network Traffic: measures total traffic of the network. Each and every packet that passes over every link is counted and added in this metric. If a packet goes from a node A to a node D passing through nodes B and C, it will increase the bandwidth usage for all three links it goes through (from A to B, B to C and then C to D).

Routers Inner and Cross Traffic: we use these two metrics to evaluate the intra-AS traffic and the traffic between ASes. The idea is the same used for the Total Network Traffic, the only difference is the separation between traffic inside the same AS and traffic between two different ASes.

C. *Simulation Results*

Preliminary results with small scenarios show that Flow Count Strategy has results very similar to the Greedy strategy. After preliminary tests we thought about running the simulations with a bigger and more representative scenario. Thus a ProwGen scenario was created. This larger scenario simulates two different flash crowds with different content objects in a 24 hours time window. Considering that HotSpot had the worst results during preliminary results we focused on analyzing the Greedy and FlowCount strategies. As expected, the results followed the same tendency of the previous scenario as depicted in Figure 2 and Figure 3.

One of the most remarkable results was the performance of the Flow Count during the flash crowds. There were two flash crowds events during the simulation. The first lasted 2-hours and was started at the 5^{th} simulation hour and the second one spent 1.5-hour and initiated at the 12^{th} simulation hour. As we can see in Figure 3, the Flow Count strategy had less cross traffic during that period. Hence the Flow Count replica server realocation outperformed the Greedy strategy during the two flash crowds events when considering the generated cross traffic. This is very interesting due to the fact that cross traffic is considered to be more expensive than traffic within a domain [15][16] and with a considerably smaller difference in total network traffic, this result demonstrates that Flow Count could be an important player in reducing network costs.

The 95^{th} percentile shows no significant difference for total network bandwidth timeline. Nonetheless, when considering cross traffic the difference is even bigger than in the previous scenario, as we can see in TABLE II. We expect great operational cost reduction when using Flow Count strategy instead of the Greedy placement strategy.

VI. LESSONS LEARNED

The first lesson learned was the need for careful selection of abstraction levels used. Our simulator, P2PCDNSim, provided much appropriate levels of abstraction, including the study of novel algorithms in the various layers of a CDN evaluation scenario, while still serving as an effective simulator for CDN networks. The second lesson learned was the need for dynamic adaptation of working conditions. Finding the best location of each replica server is important for the optimal functioning of a CDN while assuring the best QoE and network utilization metrics. This ideal location, however, varies with time, and while an adaptation is important, an instantaneous re-adaptation might lead to instability in the location of servers (for instance, the same server being relocated every 2 minutes). Therefore, a certain stability of reallocations is desired, providing a theoretically sub-optimal solution while still providing lower cross-traffic.

TABLE I. 24-HOUR PROWGEN SCENARIO

	Basic Workload	Flash Crowd 1	Flash Crowd 2
Duration	24 hours	2 hours	1,5 hours
# of objects	432	1440	810
Peer arrival	4 peers/second	40 peers/second	60 peers/second
# of requests	345512	287999	323937
Objects Popularity	Zipf (0.6)	Zipf (1)	Zipf (1)
Cacheable	40%	10%	15%
ASId	All of them	AS0	AS5
Start time	Simulation hour 0	Simulation hour 5	Simulation hour 12

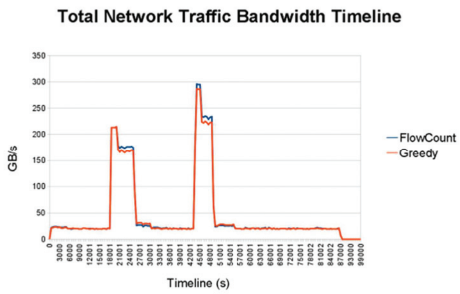

Figure 2. Bandwidth timeline during simulation for ProwGen scenario

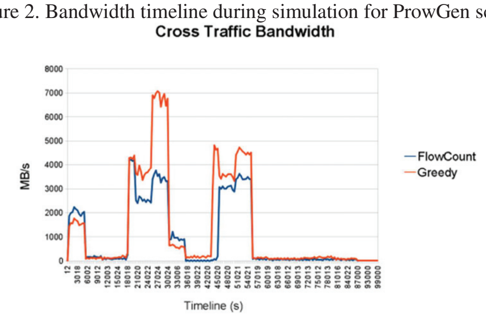

Figure 3. Cross traffic timeline during simulation for ProwGen scenario

TABLE II. 95TH PERCENTILE FOR PROWGEN SCENARIO

	Bandwidth Timeline (GB/s)	Bandwidth CrossTraffic (MB/s)
FlowCount	224,8	3516,6
Greedy	217,8	5943,7

VII. CONCLUSIONS

This work details how complex and diversified can be the evaluation of a CDN utilization scenario. This complexity, by itself, justifies the creation of a simulator to analyze CDN performance under a given configuration. The simulator developed, P2PCDNSim, was utilized in this work to analyze a novel RPA, called Flow Count, and compare it to other algorithms in the literature, namely Greedy and Hotspot.

From the results, it is clear that the novel strategy proposed, provides similar QoE in the startup delay metric with only a small increase in total network traffic. But, comparing the cross traffic it provides much better results, with significantly lower cross traffic during flash crowds, providing a good incentive for the actual deployment of the RPA in present CDNs.

Several future works can be devised. The first of them is a Replica Placement Algorithm that can also decide the number of surrogates to be placed. This is due to a limitation in all present algorithms that only deal with the placement of the surrogates, but do not estimate their appropriate number. Another future work is the evaluation of a multi-CDN scenario, where different CDNs are deployed in the same network and may compete or cooperate according to their business model.

REFERENCES

[1] K. Pathan, and R. Buyya, "A Taxonomy and Survey of Content Delivery Networks," Tech Report, Univ. of Melbourne, 2007

[2] Vakali and G. Pallis, "Content delivery networks: Status and trends," IEEE Computer Magazine, vol. 7, no. 6, Nov./Dec. 2003, pp. 68-74.

[3] Medina, A. Lakhina, I. Matta, and J. Byers. "BRITE: An approach to universal topology generation". In Proceedings of MASCOTS, Cincinnati, OH, Aug. 2001.

[4] Busari, M. and Williamson, C. Prowgen: A synthetic workload generation tool for simulation evaluation of web proxy caches, Computer Networks, vol. 38, pp. 779-794, 2002.

[5] F. L. Presti and C. Petrioli, "Distributed Dynamic Replica Placement and Request Redirection in Content Delivery Networks", 15th MASCOTS, Oct. 2007.

[6] F. L. Presti and N. Bartolini, "Dynamic Replica Placement and User Request Redirection in Content Delivery Networks", IEEE ICC'05, vol. 3, pp. 1495-1501, May 2005.

[7] Jing Sun, Suixiang Gao, Wenguo Yang, Zhipeng Jiang, "Heuristic Replica Placement Algorithms in Content Distribution Networks", in Journal of Networks, Vol 6, No 3 (2011), 416-423, Mar 2011

[8] Lili Qiu and V. N. Padmanabhan, "On the Placement of Web Server Replicas", in Proceedings of INFOCOM'01, vol. 3, pp. 1587-1596, April 2001.

[9] Mengkun Yang and Zongming Fei, "A Model for Replica Placement in Content Distribution Networks for Multimedia Applications", IEEE ICC'03, vol. 1, pp. 557-561, May 2003.

[10] P. Krishnan, D. Raz, and Y. Shavitt, "The Cache Location Problem," IEEE/ACM Transaction on Networking, Vol. 8, No. 5, 2000.

[11] S. U. Khan, "Robust CDN Replica Placement Techniques", IEEE International Symposium on Parallel and Distributed Processing, pp. 1-8, May 2009.

[12] Y. Bartal, "Probabilistic Approximation of Metric Space and its Algorithmic Applications," In Proceedings of 37th Annual IEEE Symposium on Foundations of Computer Science, October 1996.

[13] Y. Chen, R. H. Katz, and J. D. Kubiatowicz, "Dynamic Replica Placement for Scalable Content Delivery," In IPTPS 02, LNCS 2429, Springer-Verlag, pp. 306-318, 2002.

[14] Zhiyong Xu and Laxmi Bhuyan, "Qos-Aware Object Replica Placement in CDNs", in Proceedings of IEEE GLOBECOM'05, vol. 2, pp. 862-866, Dec. 2005.

[15] Ruan B, Xiong W, Chen H, Ye D. Improving Locality of BitTorrent with ISP Cooperation. 2009:2-6.

[16] J.Seedorf, S. Kiesel, and M. Stiemerling, "Traffic Localization for P2P Applications: The ALTO Approach," Proc. IEEE P2P 2009, Seattle, WA, USA, Sept. 2009.

[17] Mckeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, s., and Turner, J. OpenFlow: Enabling innovation in campus networks. ACM SIGCOMM, April 2008.

[18] Niculescu, D., Stanciu, M., Vochin, M., Borcoci, E., Zotos, N. "Implementation of a Media Aware Network Element for Content Aware Networks". CTRQ 2011, April 2011 - Budapest, Hungary.

[19] M. Rodrigues, M. Neves, J. Moreira, D. Sadok, A. Callado, P. Karlsson, V. Souza. "On traffic locality and QoE in hybrid CDN-P2P networks". Proceedings of the 44th ANSS. Boston, Massachusets, 2011.