A Model for Replica Placement in Content Distribution Networks for Multimedia Applications

Mengkun Yang Zongming Fei
Department of Computer Science, Universtiy of Kentucky
Lexington, KY 40506-0046
Email: {myang0,fei}@cs.uky.edu

***Abstract*— The distribution of multimedia files brings new challenges to the problem of replica placement in content distribution networks (CDN) and invalidates several assumptions underlying the existing solutions. In this paper we formulate a new model for the problem of replica placement to accommodate these new characteristics. We perform a theoretical analysis of the cost of distributing multimedia files over CDNs and find out that, contrary to the intuition, deploying as many replicas as possible is not always a good strategy. We then propose several replica placement algorithms that can determine the optimal number of replicas we should select from a given set of potential sites. By simulation we demonstrate that the performance of clients may degrade if we choose too many sites for replica placement.**

I. Introduction

The Internet has experienced an explosive growth over the last decade, with the web on the central stage. Making information available to a rapidly growing user population with a high service quality is quickly becoming a very important and challenging problem. The *Content Distribution Network* (CDN) has been used to push the content from the origin server to geographically distributed replicas, which bring content to the edge of the network where the clients are attached. One of the key decision problems is replica placement, which selects a subset from a group of potential sites to put replicas, given the knowledge of client access patterns. Replica placement is usually done by a system administrator and many CDN service providers tend to acquire as many replication sites as possible. Recently we have seen proposals giving algorithmic solutions to these problems [1], [2]. However they target the CDNs for delivering traditional web pages, such as small image or html files. We found that several assumptions underlying these algorithms do not hold any more for multimedia applications.

One assumption is that the storage at a replica is large enough to hold any number of objects and there is no need to replace an item in the storage. However the task of distributing multimedia files brings new changes to CDNs. Objects are usually much larger and thus replicas can no longer hold all the files. For example, an MPEG movie of 2 hours can be as large as 1.35GB and a 100GB disk can only hold less than 75 files. We need to reconsider the assumption when we study the problem of replica placement for multimedia applications.

Another assumption is that the cost of delivering documents from replicas to clients should be the focus and the cost of distributing documents from origin servers to replicas are of secondary concern. However the distribution of multimedia files requires higher bandwidth and usually lasts for a long period of time. As CDNs develop, more and more replicas are now established. More importantly, content distribution internetworking (CDI) [3], [4] tends to connect CDNs together. Therefore the number of replicas which the objects at the origin server will be delivered to can be very large. The traffic for distributing objects will be much heavier and can no longer be ignored.

To deal with these new circumstances in the content distribution networks for multimedia applications, we formulate a new model to reflect new characteristics of the replica placement problem. The basic assumption is that the storage size at a replica is limited and more importantly, the distribution of a document from the origin server to a replica is compensated by a finite number of requests from clients to the replica. We derive a formula to calculate the hit ratio in this finite capacity and finite access scenario. We make a simple hypothesis about the relationship between the distances from a client to its nearest replica and to the origin server when the number of replicas changes. Based on it we perform a theoretical analysis of the cost of replica placement as we put more replicas over there. In particular, we are interested in finding out whether we should choose as many sites as possible to put replicas. Contrary to the intuition, we find that when the number of replicas exceeds a threshold the performance will deteriorate. We propose several replica placement algorithms that give the optimal number of replicas and result in the minimal total cost. Our simulations validate the model proposed and demonstrate that the performance of clients may degrade if we choose too many sites for replica placement.

The rest of this paper is organized as follows. We present a new model for replica placement in Section II. Based on the analysis under the model we propose algorithms for replica placement in Section III. They are followed by performance evaluation in Section IV and related work in Section V. We conclude the paper in Section VI.

II. A New Model for Replica Placement

A. Problem Description

The problem of replica placement can be described as follows. Given a network topology, the locations of clients and their request volumes, and a set S of nodes as potential

This work was supported in part by the National Science Foundation under Grants CCR-0204304 and EIA-0101242.

replica sites, find a subset of S to place replicas so that some performance measure is optimized. The goal can be simply to minimize the cost of accessing the objects from replicas by clients, or this cost plus some additional costs, such as the cost of distributing documents from the origin server to replicas, and/or the cost of maintaining replicas.

Two well-known graph problems have been used to model the replica placement problem. The first one is the *k-median problem*: Given m potential sites for replica placement and a fixed k ($k \leq m$), select k sites from them to put replicas, such that the total cost $\sum d_j c_{ij}$ is minimized, where d_j is the demand from client j and c_{ij} is the cost (or distance) from client j to the nearest replica (assume it is replica i).

Another one is the *facility location problem*: Given m potential sites for replica placement, find a subset from m sites to put replicas, such that the total cost $\sum f_\ell + \sum d_j c_{ij}$ is minimized, where f_ℓ is the cost of setting up replica ℓ, d_j is the demand from client j and c_{ij} is the cost (or distance) from client j to the nearest replica (assume it is replica i). We can find an optimal value for the number of replicas (assume it is k) in this model.

Most of current work [1], [2] uses k-median problem as a model for replica placement. The basic assumption is that each replica can hold all the contents from the origin server. As we pointed out previously this is not true in the multimedia environment because of the large size of multimedia files. Therefore contents in a replica may be replaced by new ones in the future. The model does not include the cost of distributing objects from the origin server to replicas into the picture. As the number of replicas become larger and larger, this traffic cannot be ignored, especially for multimedia objects which require a high bandwidth for a long period of time for its distribution. The model requires a pre-determined k for the number of replicas. It is not obvious how to choose an appropriate value for it.

While the facility location problem will generate the optimal value for k, the problem with the model is that it is hard to define appropriate values for the cost of setting up a replica (f_ℓ). In our view, this cost should not be a static value for each replica, but rather it depends on the access pattern from clients and how many times the same copy of an object has to be delivered from the origin server to replicas. The facility location problem cannot capture this dynamic aspect of the performance involved in the content distribution.

We give a new formulation of the problem. Given m potential sites for replica placement, find a subset to put replicas, such that the total cost $\sum(d_j * hit\_ratio * c_{ij} + d_j * (1 - hit\_ratio) * (c_i + c_{ij}))$ is minimized, where d_j is the demand from client j, c_{ij} is the cost (or distance) from client j to the nearest replica (assume it is replica i) and c_i is the cost (or distance) from replica i to the origin server. There exists an optimal value for the number of replicas such that the total cost is minimized. Note the cost in the model can be latency, hop counts, etc. It can also be modified to represent the traffic generated for delivering objects to clients if we assume that all objects are of equal size.

We introduce the hit_ratio into the formula for the optimization and categorize two kinds of accesses, those getting a copy of the object from the replica (by $d_j * hit\_ratio * c_{ij}$) and those getting a copy from the origin server (by $d_j * (1 - hit\_ratio) * (c_i + c_{ij})$). They have different costs. If it is a hit, the cost is c_{ij}, the distance from the client to the nearest replica; if it is a miss, the cost is $c_{ij} + c_i$, the distance from the client to the replica plus the distance from the replica to the origin server.

Similar to the facility location problem we believe k should not be pre-determined and there is an optimal k that leads to the best performance. Here are our observations. If we only have a very small number of replicas, clients will take a long path to get the requested object from one of them or the origin server. This will result in the long response time at the clients, as well as the high load on the network and the origin server. However, if the number of replicas is too large, the hit ratio will be small and the cost of delivering objects from the origin server to replicas will be shared by fewer client requests. Therefore the total cost will increase. At the same time, too many replicas result in substantial load for delivering objects to replicas and saturate the network with delivery and updates. This will potentially hurt the response time of clients.

B. Hit Ratio

Key to the problem formulation is how to model the hit ratio. Assume the total number of files is N, the size of storage at a replica is C and the total number of requests to these N files associated with a given replica is R. We assume all objects are of equal size and C is the number of files that can be stored at a replica. Previous work [5] demonstrated that the web page requests follow the Zipf-like distribution. Assuming the objects are ranked from 1 to N based on their popularity (1 means most popular), the probability of requesting object i is proportional to $\frac{1}{i^\alpha}$, where $0.65 < \alpha < 0.85$. More recent results [6] found that the distribution of accesses to multimedia documents is also Zipf-like with $\alpha = 0.733$ for the trace recorded.

By the Zipf-like distribution, the probability of requesting object i is:

$$P_N(i) = \frac{\Omega}{i^\alpha}, \text{ where } \Omega = \left(\sum_{i=1}^{N} \frac{1}{i^\alpha} \right)^{-1}$$

Hit-ratios have been derived in the following two cases [5]: 1) infinite capacity C of the storage, finite number of requests R; 2) finite capacity C of the storage, infinite number of requests R. We will derive a hit-ratio for the finite capacity C and finite number of requests R, and use it in our model. It is easy to understand that the finite capacity is more realistic in the multimedia environments, as explained previously. Actually we found that using finite number of requests R in model also better reflects the reality. Observe that all documents are accessed only for a certain period of time, though the life time may vary from one document to another. After some time, the current set of documents in a replica will not be

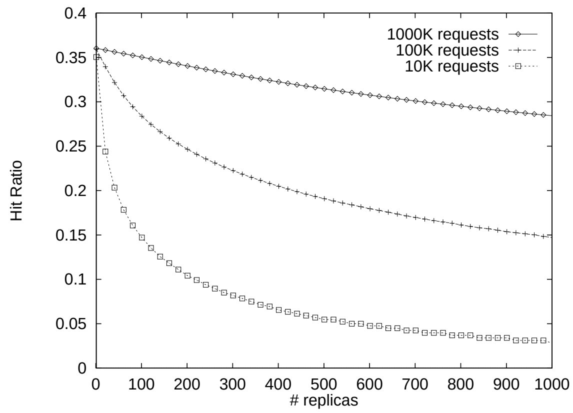

Fig. 1. Hit ratio

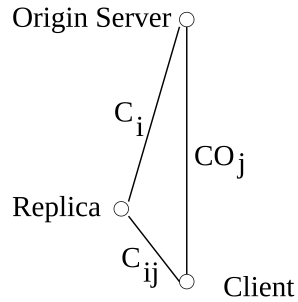

Fig. 2. The relationship between distances

accessed at all. It is beneficial to replace them with a new set of documents. We can see that the cost of distributing a document from the origin server to a replica is only compensated by a finite number of requests from clients to a replica.

To simplify the derivation, we assume only the most popular C objects are stored in the replica. Assuming that requests are independent, we consider the hit ratio at the j-th request, where $1 \leq j \leq R$. The probability of accessing page i is $P_N(i)$. During previous $j-1$ requests, the probability of accessing page i is $1-(1-P_N(i))^{j-1}$. Therefore the probability of a hit at j-th request is:

$$H_N(j,C) \approx \sum_{i=1}^{C} P_N(i)(1-(1-P_N(i))^{j-1})$$

The average hit ratio over R accesses is:

$$H(R,C,N) = \frac{1}{R}\sum_{j=1}^{R} H_N(j,C)$$

That is,

$$H(R,C,N) = \frac{1}{R}\sum_{j=1}^{R}\sum_{i=1}^{C} P_N(i)(1-(1-P_N(i))^{j-1})$$

We found that with given N and C, the hit ratio decreases as R decreases, i.e., the smaller number of requests will result in a smaller hit ratio. With more replica servers, the number of requests to each replica will decrease, which will result in a smaller hit ratio. In Fig. 1 we depict the hit ratio as we change the number of replicas from 1 to 1000. We have one million files and the storage at a replica can hold one hundred multimedia files. The total number of requests to all replica servers is either 10K, 100K or 1M. When the number of replica servers increases from 1 to 1000, we can see that the hit ratio decreases significantly, especially in the case when the total number of requests is small.

C. Cost of Placement

For a given placement, we can calculate c_{ij}, the distance for client j to its nearest replica, and c_i, the distance from replica i to the origin server. To make a theoretical analysis of the total cost, we make an assumption about the relationship between the average distance from clients to origin servers and the average distance from clients to their nearest replicas. Based on it, we derive a model for the total cost of replica placement changing as the number of replicas increase.

Assume the average distance from clients to origin servers (over all client, origin server pairs) is r. As we put more replicas, the average distance from clients to their nearest replicas will decrease. It can be modeled as a function of r and the number of replicas k, denoted as $f(r,k)$. We make a hypothesis about the relationship by defining $f(r,k) = \frac{r-1}{\beta^{k-1}} + 1$, where β is a decay factor and $\beta > 1$. For a network with a few hundreds of nodes, β will be large because increasing the number of replicas will reduce the distance from a client to its nearest replica significantly. For a network of the Internet size, β tends to be very close to 1, but it is still an exponential decrease when we increase the number of replicas.

Based on the above model, we analyze the total cost of replica placement when the number of replicas changes. In Fig. 2 we illustrate the relationship between three distances: c_{ij}, the distance from a client to its nearest replica, c_i, the distance from the replica to the origin server, and co_j, the distance from the client to the origin server. We know co_j can be modeled by the average distance r and c_{ij} can be modeled by the function $f(r,k)$. The other term used in calculating the total cost is $c_{ij} + c_i$. Usually replicas are located at access points of clients to the network and therefore we have $c_i \leq co_j$. Combining it with the triangle relation, we have $co_j \leq c_{ij} + c_i \leq co_j + c_{ij}$. As we put more replicas, the replica is more likely to be located on the path from the origin server to the client and therefore $c_{ij} + c_i \approx co_j$. With this simplification we can use r to model the cost $c_{ij} + c_i$.

We are specifically interested in the relationship between the total cost of replica placement and the number of replicas. Thus we ignore the differences between the demands from different clients and assume they are equal. To make the cost of placement in different request volumes comparable, we plot the average cost rather than the total cost. In Fig. 3, we assume the total number of objects is 1 million, the average distance r equals 15 and decay factor $\beta = 1.01$. The number of replicas is changed from 1 to 1000. We plot three cases with the total number of requests equals 10K, 100K, 1M, respectively. We

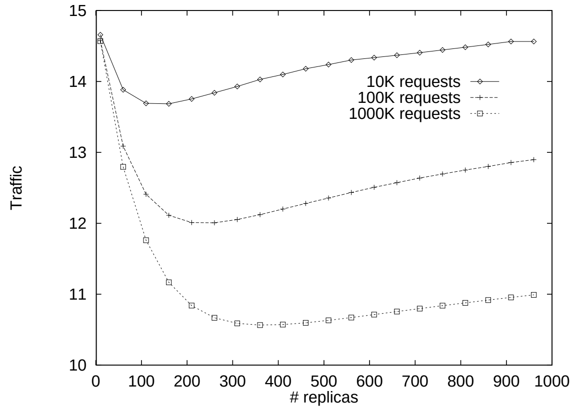

Fig. 3. The cost

can find that the cost will increase after the number of replicas reach a certain threshold, which depends on the total number of requests. Actually this threshold will also change when the network size or the total number of objects change.

III. Algorithms for Replica Placement

Based on the above analysis, we can expect a V-shape in the plot of the cost versus the number of replicas. Instead of iterating over all possible k values, we can get the optimal number of replicas by observing the cost generated. We find that those heuristics, including greedy [1], hot spot [1], and max fanout [2], can be modified for generating a solution to our formulation with a best k, because they select one replica at a time until the number of replicas reaches the pre-determined k. For example, we can modify the greedy heuristic to add one replica at a time using our performance measure. First it decreases as the number of replicas selected increases. To a certain point, this measure begins to increase. That's where we will find the optimal k value.

A. Greedy Algorithm

The greedy algorithm chooses one replica at a time until it observes an optimal solution. At the first step each of the N potential sites is evaluated individually. For each site we assume all clients will get the object from it. We calculate the distance from each client to the site (denoted as c_{ij}). Based on the hit ratio calculated in the previous section we can get the total cost $\sum(d_j * hit\_ratio * c_{ij} + d_j * (1 - hit\_ratio) * (c_i + c_{ij}))$, where c_i is the average distance of the replica site to all origin servers. We pick the replica with the minimal total cost, denoted as TC_1. At the second step we select the second replica among the remaining replica sites such that when it is combined with the first replica we selected, the total cost is minimal. When calculating c_{ij} we use the distance to the nearest replica sites selected. The total cost is denoted as TC_2.

This process continues until we find a j such that $TC_{j-1} < TC_j < TC_{j+1}$. We get the total cost of TC_j and the number of replica sites is j.

B. Hot Spot Algorithm

The hot spot algorithm differs from the greedy algorithm only in selecting the best candidate in each steps. First for each replica site the traffic generated within its vicinity is calculated as the total number of requests from clients within a radius. At the first step select a replica site with the maximal number of requests (hot spot) and calculate the total cost TC_1 in the same way as the Greedy algorithm. At the second step select the next hottest replica site and calculate the total cost TC_2. This process continues until we find an optimal solution.

C. Max Fanout Algorithm

The max fanout algorithm selects a replica site with the maximum fanout at each step and calculate the total cost. The process stops when the optimality condition is satisfied.

One problem we want to discuss is the local optimal. It is possible that the algorithm stops when the optimality condition is satisfied, but it is not the global optimal but a local optimal. One remedy measure is to continue the process until $2 * j$ to reduce the possibility of the local optimal. We will illustrate the phenomena in the simulation section.

IV. Performance Analysis

To study the effect of changing the number of replicas on the performance defined in the new replica placement model in Section II, we conduct simulations on network topologies generated by the GT-ITM Internet topology generator [7], [8]. We generate hierarchical graphs by using the transit-stub model in GT-ITM. In this model, topologies representing connected transit domains, connected transit nodes, and connected stub domains are constructed in turn, and finally additional edges are added if necessary.

For this study, we generate 10 Transit-Stub graphs with 600 nodes each. Each generated topology represents a network with three transit domains, and three stub domains per transit node; on average, each transit domain, as well as each stub domain has 8 nodes.

In this simulation, we place the server on a stub node. One selected node in each stub domain is designated as a potential replica site. In addition to the nodes already chosen as the server and replica candidates, we select 25% nodes in each stub as clients. The requests to the origin server are uniformly distributed among all the clients. The probability of an object in the origin server being accessed follows the Zipf-like distribution [5], [9].

Various methods for choosing k replica sites among m candidates are evaluated: (1) greedy algorithm [1], where in each iteration, a candidate site which, combined with the sites already picked, yields the lowest cost, is picked to place replica, (2) hot spot algorithm [1], where replicas are placed at sites with the greatest number of demands in its vicinity with a certain radius, and (3) maximum fanout algorithm [2], where the sites with largest fanouts are chosen as replica sites. For each value of k, we use above algorithms to choose replica sites, and then compute the average latency of clients. Instead of stopping at the optimal k, we continue the process to the

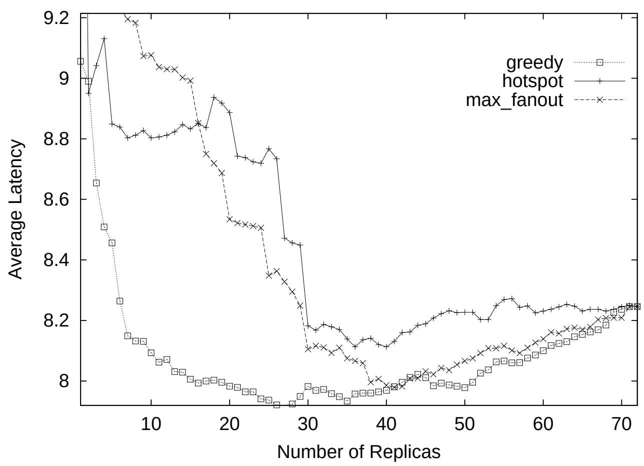

Fig. 4. Average latency using LRU

maximal possible number of replicas. We assume every replica site has a limited storage and experiment with two replacement algorithms. One is the well-known Least Recently Used (LRU) and the other is the Most Popular (MP) algorithm, which never removes N most popular objects from the storage, and replaces other objects based on LRU.

Figure 4 shows the impact of changing k on the average latency when we use LRU as the replacement algorithm. When the number of replicas increases, the average latency first decreases because having more replicas makes the distance from a client to the nearest replica shorter and the average latency therefore decreases for all three placement algorithms. However, when the number of replicas increases beyond 30 (in the case of the greedy algorithm), the latency begins to increase. This is also true for using max fanout and hot spot algorithms. As we can see, there exists an optimal value of k for all these three algorithms. We can derive from the figure that increasing the number of replicas beyond a certain value may even lower the performance.

V. Related Work

Replica placement is a recent research topic. The instrumentation placement scheme uses graph-theoretic methods and *ad hoc* heuristics to decide the locations of tracers to obtain Internet distance maps [10]. Web proxy placement uses a dynamic programming approach to determine the optimal locations of proxies in a tree topology [11]. More recently, the work on web server replica placement evaluates several placement heuristics with different topologies [1]. The topology-informed replica placement [2] uses the fanout, the number of connected Autonomous Systems (AS) to decide locations to put replicas and propose a simpler algorithm with comparable performance. All these schemes formulate the placement problem as "selecting k locations from m possible locations" and do not consider the cost for delivering objects from the origin server to replicas. Our replica placement model includes this cost and can generate an optimal k for the placement problem. It can better deal with the future interconnection among CDNs and the reality of larger multimedia files being used in the distribution network. Constrained mirror placement scheme [12] tries to minimize the round trip time of clients and finds out that increasing the number of mirror sites is effective in reducing client download time and reducing server load only for a surprisingly small range of values regardless of the mirror placement algorithm. We go one step further and find out that too many replicas or too many copies of an object may actually hurt client performance in the multimedia environment.

VI. Concluding Remarks

In this paper we analyzed the assumptions underlying the existing models for the problem of replica placement in content distribution networks. To accommodate the new characteristics brought by the multimedia applications we propose a new model for replica placement. We perform a theoretical analysis of the cost of distributing multimedia files over CDNs and find out that, contrary to the intuition, deploying as many replicas as possible is not always a good strategy. We then propose several replica placement algorithms that can determine the optimal number of replicas we should select from a given set of potential sites. By simulation we demonstrate that the performance of clients may degrade if we choose too many sites for replica placement.

References

[1] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, "On the placement of web server replicas," in *Proceedings of INFOCOM'01*, April 2001, pp. 1587 –1596, anchorage, Alaska.

[2] P. Radoslavov, R. Govindan, and D. Estrin, "Topology-informed internet replica placement," in *Proceedings of Sixth International Workshop on Web Caching and Content Distribution (WCW'01)*, June 2001, pp. 55–70, boston, MA.

[3] A. Biliris, C. Cranor, F. Douglis, M. Rabinovich, S. Sibal, O. Spatscheck, and W. Sturm, "CDN brokering," in *Proceedings of the 6th International Workshop on Web Caching and Content Distribution*, June 2001, pp. 19–34, boston, MA.

[4] M. Green, B. Cain, G. Tomlinson, S. Thomas, and P. Rzewski, "Content internetworking architectural overview," *Internet Draft*, 2001, http://www.IETF.org/internet-drafts/draft-green-cdnp-gen-arch-03.txt.

[5] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence, and implications," in *Proceedings of INFOCOM'99*, March 1999, pp. 126 –134.

[6] D. Luparello, S. Mukherjee, and S. Paul, "Streaming media traffi c: An empirical study," in *Proceedings of Sixth International Workshop on Web Caching and Content Distribution (WCW'01)*, June 2001, boston, MA.

[7] K. Calvert, M. Doar, and E. W. Zegura, "Modeling Internet topology," *IEEE Communications Magazine*, pp. 160–163, June 1997.

[8] E. W. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proceedings of INFOCOM'96*, 1996, pp. 594 –602, san Francisco, CA.

[9] G. Zipf, Ed., *Human Behavior and the Principle of Least Effort*. Reading, MA: Addison-Wesley, 1949.

[10] S. Jamin, C. Jin, D. Raz, Y. Shavitt, and L. Zhang, "On the placement of Internet instrumentation," in *Proceedings of INFOCOM'00*, March 2000, pp. 295 –304.

[11] B. Li, M. J. Golin, G. F. Ialiano, and X. Deng, "On the optimal placement of web proxies in the Internet," in *Proceedings of INFOCOM'99*, March 1999, pp. 1282 –1290.

[12] S. Jamin, C. Jin, A. R. Kurc, D. Raz, and Y. Shavitt, "Constrained mirror placement on the internet," in *Proceedings of INFOCOM'01*, April 2001, pp. 31 –40, anchorage, Alaska.