Heuristics for the Design and Optimization of Streaming Content Distribution Networks

Sandjai Bhulai [a]
[a] Vrije Universiteit Amsterdam
Faculty of Sciences
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Email: S.Bhulai@few.vu.nl

Rob van der Mei [a,b]
[b] CWI
Advanced Communication Networks
P.O. Box 94079
1090 GB Amsterdam
The Netherlands
Email: R.D.van.der.Mei@cwi.nl

Mengxiao Wu [c]
[c] CWI
Computational Intelligence
P.O. Box 94079
1090 GB Amsterdam
The Netherlands
Email: Mengxiao.Wu@cwi.nl

***Abstract*— The design of efficient and scalable streaming Content Distribution Networks (CDNs) is an open problem of great economic interest. A key decision is the number of replica servers to use and their location to balance network bandwidth and server costs. This problem is highly complex, since it is interrelated to the streaming media delivery protocols, client redirecting decisions, and the content delivery routing problem.**

To address this design problem, we develop heuristics for the server placement, the server selection, and the routing problem. In contrast to previous work, we study both periodic broadcast and hierarchical merging protocols using multiple media files with different request rates. The heuristics lead to highly accurate and fast approximations. We also analyze the relationship between the start-up delay caused by some delivery protocols, the bandwidth consumption, and the number of replica servers, providing complete and fully implementable solutions to the design of streaming CDNs.

I. Introduction

Streaming Content Distribution Networks (CDNs) address the problem of efficient streaming media delivery through replicating media files on replica servers distributed in the network. The aim is to balance the costs of using network resources and the server costs while meeting a desired media playback quality. Although streaming CDNs are based on the traditional CDN technology, the specific characteristics of streaming media files make them different from traditional web CDNs, which have been developed for many years already (by, e.g., Akamai and AOL). In spite of the advances in research and development of streaming techniques, such as high-performance video servers and various streaming media delivery protocols, the design of efficient and scalable streaming CDNs is an open problem of great economic interest.

The support of streaming media services through streaming CDNs increases scale and reach of streaming media. However, handling streaming media files also brings additional complexities with CDN mechanisms due to their bandwidth-intensive nature of high quality and the long-life nature (tens of minutes to a couple of hours). The key questions that can be identified in the design of efficient streaming CDN design are: a) the server content problem: which content should be replicated at the proxy servers, b) the server placement problem: how many replicas are needed and where should each replica server be placed in the network, c) the server selection problem: to which server should a client's request be directed to, and d) the request routing problem: how to route the content from the selected server to the client. These questions are the focus of this paper.

A. Related literature

There is a lot of literature presenting optimal solutions or approximate algorithms for the server placement problem in traditional web CDNs [1], [2], [3], [4], [5], [6], [7], [8]. However, these models and algorithms are unfit for the design of streaming CDNs, because they do not account for the special characteristics mentioned earlier of streaming media files and multicast delivery. Generally, a greedy algorithm can provide solutions to the server placement problem in web CDNs with a performance that is close to optimal [5]. The server placement problem for streaming CDNs has only been partially addressed in a few studies [9], [10], [11], [12].

The server selection problem is addressed in [10] for a periodic broadcast or live broadcast system in which the network bandwidth is independent of client loads. Both [9] and [10] address the case in which the system has only one server. In [11], a logarithmic approximate algorithm and a heuristic solution are presented to construct application-layer multicast trees with the target of minimizing the delay.

B. Contribution

The server placement, selection and routing for streaming CDNs that use scalable streaming media delivery approaches is further explored in this paper. Our work partially builds on the results from [12], and proposes a set of optimization models and heuristic algorithms for both placement and routing. The following are the distinguishing points.

We address the problem for streaming CDNs using not only hierarchical streaming merging protocols, but also periodic broadcast protocols. In contrast to [9], [10], we also discuss algorithms for the case of multiple servers, as Almeida [12] does. However, Almeida only considers the case of a single media file. In this paper, we also deal with the case of multiple media files with different request frequencies, and develop algorithms that determine the placement and routing for multiple media files concurrently in streaming CDNs with total delivery costs that are closer to optimality.

Our cost function is different from the cost function used in [12], in the sense that it explicitly takes into account the trade-off between deploying more replica servers to save

network bandwidth and having less replica servers to reduce server costs. Based on this cost function, we provide heuristics and study the relationship between the start-up delay caused by some delivery protocols, the bandwidth consumption, and the number of replica servers. The heuristics are shown to be fast, yet highly accurate, and easily implementable, and well-suited to be used in the design and optimization of commercial CDN infrastructures.

II. Problem formulation

The goal of streaming CDNs is to provide the best available performance to its clients while placing as few replicas as possible. In this section, we develop models to determine the optimal replica placement and delivery trees such that network costs are minimized. To this end, consider a given network topology (V, E) with $V = \{1, 2, \ldots, n\}$ a set of nodes and $E \subseteq \{(i, j) \,|\, i, j \in V\}$ a set of undirected edges. We assume that each edge $(i, j) \in E$ has a weight w_{ij}, representing the number of router hops on each edge having equal costs per unit of bandwidth. Furthermore, let $C^f \subseteq V$ denote the set of client nodes for various multimedia files $f \in F$, and $S \subseteq V$ the set of server nodes with $S \cap C^f \neq \emptyset$. We assume that each file f has a start-up delay of d, a duration of T (in minutes), and that the set of client loads that arrives during the playback is L_i^f for $i \in C^f$ and $f \in F$.

We first deal with the placement of a replica on a single server node. A server node with requests acts as a client node when no replica is placed on it. All router nodes and other server nodes without requests are linked optionally as determined by the optimization requirement. The flow X_{ij}^f over an edge $(i, j) \in E$ for file $f \in F$ is defined as the sum of all client loads for the same file U^f that share that edge in the delivery tree. The usage Y_{ij}^f of an edge indicates whether the edge belongs to the delivery tree or not.

The objective is to determine where to place a replica such that the network costs are minimized. The network costs are defined as the proportional value of the network bandwidth consumption. Note that these costs depend on the media delivery protocol that is used in the network. Periodic Broadcast (PD) protocols aim to achieve short start-up delays by frequent broadcasts of the initial segments of the media file. Low bandwidth consumption is obtained by making broadcasts of later segments less frequent. Hu [13] shows that the required bandwidth of PD protocols only increases logarithmically as $\ln(T/d + 1)$. Hence, the networks costs are given by

$$B_{\text{network}} = \ln\left(T/d + 1\right) \sum_{f \in F} \sum_{(i,j) \in E} w_{ij} Y_{ij}^f.$$

Periodic broadcast protocols have the drawback that they have a start-up delay. This lead to the development of the Hierarchical Multicast Stream Merging (HMSM) protocol. This protocol aggregates clients that request the same media file into successive groups. Upon receiving a client request for a media file, the server initiates a multicast stream delivering content from the beginning of the file, so that the client can start playback immediately. Simultaneously, the client snoops on the closest earlier stream which is delivering the same file. The client's own stream is terminated when the client has received all file data prior to that obtained by snooping. Under this protocol, the network costs are given by (see [14])

$$B_{\text{network}} = \sum_{f \in F} \sum_{(i,j) \in E} w_{ij} \ln(X_{ij}^f + 1).$$

When the broadcast protocol has been fixed, the replica placement problem can be cast as a mixed integer programming problem given by

$$\begin{aligned}
&\text{min } B_{\text{network}} \text{ subject to} \\
&\sum_{i \,|\, (i,k) \in E} X_{ik}^f - \sum_{j \,|\, (k,j) \in E} X_{kj}^f = L_k^f, && k \in V, f \in F, \\
&X_{ij}^f \leq U^f Y_{ij}^f, && i, j \in E, f \in F, \\
&U^f = \sum_{i \in V} L_i^f, && f \in F, \\
&X_{ij}^f \geq 0, && i, j \in E, f \in F, \\
&Y_{ij}^f \in \{0, 1\}, && i, j \in E, f \in F.
\end{aligned}$$

In our opinion, this model is more realistic than the models proposed in [12], since our model computes that total bandwidth consumption during the complete playback period. Also note that the problem for a reasonable instance is quite large and has long computation times. In practice, only small instances can be calculated to optimality using this problem formulation. Therefore, one needs to take refuge to heuristics for obtaining solutions for realistic problem instances.

III. Heuristics for the server placement

In this section we present heuristics to solve the server placement problem for streaming CDNs. We first start with the single server placement problem. The solution to this problem consists of choosing the optimal server node to place the replica on, and of building the delivery tree connecting all client nodes. Thus, the solution can be seen as a routing and placement pair. We present accurate algorithms to these separate problems that require significantly less execution time than solving the combined problem presented in the previous section. Therefore, they can be applied to large and scalable systems with nearly optimal delivery costs.

We first focus on routing heuristics for the server placement. In the sequel we shall distinguish between the network costs for the PD and the HMSM protocol. We shall refer to the costs of the former as the network distance, and to the latter as network costs. The heuristic starts by building a tree of shortest paths. This can in principle be used for the routing (the shortest path routing SP). However, the heuristic uses incremental costs/distance to create better schemes GC/GD for routing. The algorithm is as follows.

Prime routing heuristic

1) For a given topology with $|V|$ nodes, use each node $n_i \in V$ as the root to build a tree of shortest paths sp_i to connect all other $|V| - 1$ nodes.
2) For a given server node $s_j \in S$, which has placed or is supposed to place a replica on it, add it to the delivery tree dt as the first and root node.
3) Choose a client node $c_k \in C$, which is yet unconnected by dt, and is supposed to be connected to the closest node n_i that is already in dt via the tree sp_k. Then, calculate and store

the incremental network costs/distance (for GC/GD) of this (temporary) delivery tree.
4) Repeat step 3 until all unconnected client nodes are exhausted. Add client node c_k, with the least incremental network costs/distance, and the shortest path from it to n_i on the sp_k into dt. If there is a router node on this selected shortest path, add it into dt as part of the selected shortest part.
5) Repeat step 3 and 4 until all client nodes have been included in dt. The delivery tree dt presents the routing decision tree.

The three algorithms build a delivery tree given the decision of the replica. We base our placement heuristic on these three algorithms and combine them with network costs/distance to finally derive four placement heuristics GCC (GC with network costs), GDD (GD with network distance), SPC (SP + network costs), and SPD (SP + network distance). The algorithm works as follows.

Prime placement heuristic

1) For each server node $s_j \in S$ build a delivery tree dt_j, which is rooted in s_j and connects other nodes by using the GC/GD/SP heuristic. Calculate and store the network costs of dt_j.
2) Select the server node that has the minimum network costs/ distance for the replica placement.

The two previous algorithms solve the problem posed in Section II resulting in four algorithms SGCC (GC + GCC), SGDD (GD + GDD), SSPD (SP + SPD), and SSPC (SP + SPC). Experiments show that the SGCC/SGDD heuristics perform very well and are close to optimal. Since the SP routing algorithm is much faster, the SSPC/SSPD algorithms are much faster as well.

Let us now focus attention on the placement of multiple replicas. In this case, one needs to solve the client assignment problem as well as the routing and placement problem. However, these problems are related to each other, and increase the complexity of the problem significantly. The following heuristic deals with this interaction as follows.

Heuristics based on the greedy algorithm

1) For a given topology tp with $|V|$ nodes and N replicas, use each server node $s_j \in S$ as the root for a shortest path tree sp_j.
2) Choose an unselected server node s_j and place a (temporary) new replica r_j on it. Connect each client node $c_k \in C$ to the nearest replica r_l (among existing replicas and r_j) via sp_l. This determines the sub-topology tp_l.
3) Use the replica r_l as the root to implement the GCC/GDD/ SPC/SPD heuristic on tp_l to get the network costs/distance of tp_l. Compute the total network costs/distance by adding up the costs of each sub-topology, and store this.
4) Repeat step 3 and 4 until all unselected server nodes are exhausted. Place a new replica on the server node with the least incremental network costs/distance.
5) Repeat step 3 and 4 until all N replicas are placed.
6) Assign each client node to the nearest replica via the shortest path computed previously, and use GC/GD/SP to build the delivery tree.

The heuristic results in four algorithms MG-GCC, MG-GDD, MG-SPC, and MG-SPD. Although the accuracy of the above algorithm might be high, its computation time is also quite large. To reduce the computation time, we can partition or cluster the system based on the K-means algorithm [15] and solve the problem in each subsystem. This results in four new heuristics MC-GCC, MC-GDD, MC-SPC, and MC-SPD as follows.

Heuristics based on the clustering algorithm

1) For a given topology tp with $|V|$ nodes and N replicas, use each server node $s_j \in S$ as the root for a shortest path sp_j.
2) Select N nodes randomly among the server nodes to place the replicas.
3) Connect each client/router node n_i to the nearest (temporary) replica r_j via the shortest path sp_j. This determines the sub-topology tp_j.
4) For each sub-topology tp_j, iteratively use a replica r_j as the root to implement the GC/GD/SP routing heuristic to construct a tree connecting all client nodes in thus sub-topology. Calculate the network costs/distance, and select the node with minimum network costs to place the replica on.
5) Repeat step 3 and 4 until the maximum number of iterations has been reached, or when the N selected replica servers do not change. After the loop, place the replicas on the last set of selected nodes.
6) Use the GC/GD/SP routing heuristic to construct the delivery trees with the N determined sub-topologies and their own replicas.

IV. Numerical experiments

We have implemented the heuristic algorithms on a 50-node example network. In this study, we used the Georgia Tech Internetwork Topology Models software GT-ITM [16], [17] to generate underlying network topologies. This software contains routines that generate networks based on a variety of network models. In particular, we use two flat random network models, the Pure Random (PR) model and the Waxman model [18], and one hierarchical Transit-Stub (TS) model [16].

In our examples, we assume that the network topology is fixed. Consequently, the management of a multicast group does not involve client nodes that join or leave the multicast group. The multicast functionality at the network layer is still not widely deployed in IP networks. Additionally, it is widely accepted that the shortest path tree, which is adopted by various multicast routing protocols, is not a good solution for constructing optimal delivery trees in streaming CDNs [19]. Therefore, we assume that (nearly) optimal constructions of delivery trees are implemented by the application layer multicast (see [20]).

We use three multimedia files with different characteristics. The client requests for each file are homogeneous with $L^1_{C^1} = 10$, $L^2_{C^2} = 1$, and $L^3_{C^3} = 0.1$. Furthermore $(S, C) = (25, 20, 10, 5)$ for the PR and Waxman topology, and $(S, C) = (12, 30, 20, 10)$ for the TS topology.

We focus on a two-level hierarchical architecture in which clients send requests to the original server or the replica servers. The client is only served only by replica servers, thus the original server does not transmit media contents directly to the client. This shifts the problem to the optimal placement of replica servers. Moreover, we do not assume a peer-to-peer architecture in which transmission between replica servers or between client nodes is possible. Thus, each client node is limited to get service from only one replica server, which get its media contents from the original server.

In general, the replica server can choose to either fully or partially replicate the multimedia file. However, we assume that the file is fully replicated, since [12] shows that full file replication outperforms prefix or partial replication in a very large region of the design space. We suppose that the replica server can only run one type of delivery protocol (PB

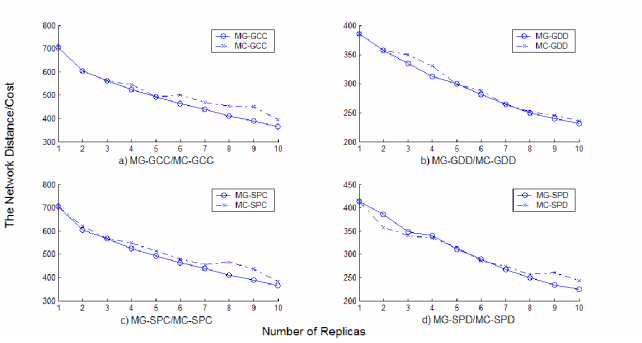

Fig. 1. Network costs/distance for the PR network

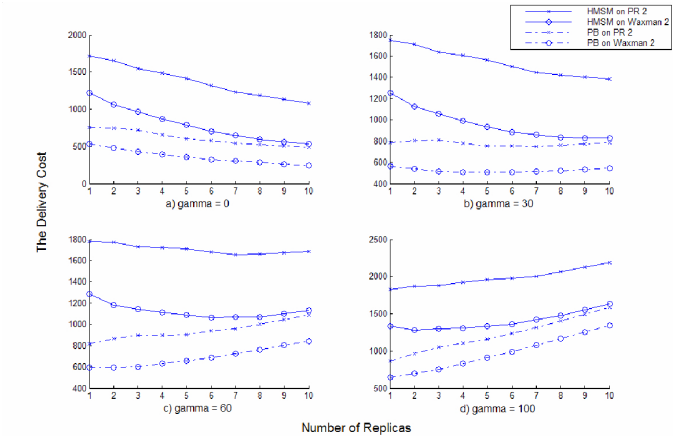

Fig. 2. Delivery costs vs. number of replicas for different values of γ in PR and Waxman networks with $\alpha = 0$ and $\beta = 1$.

or HMSM). However, in the case with multiple replica servers, we will consider the case where part of the servers run the PB protocol, and the others run the HMSM protocol.

The results for the PR network are shown in Figure 1. The results for the other networks are similar. The experiments show that the heuristics have comparable performance with the greedy algorithm being slightly better. The biggest gap between the two heuristics occur under the PR network. The heuristic based on the clustering, however, has a very high efficiency as the computation times become shorter when the number of replicas increases. Hence, in large scale environments preference is given to this algorithm.

V. Issues in the Design of Streaming CDNs

In the previous section, the heuristic algorithm ignored the protocol costs, the server costs, and the incremental network bandwidth (under the PB protocol) for the sake of accurate and efficient solutions to the server placement problem. In this section we discuss the trade-offs between the start-up delay, the number of replicas, and the network bandwidth based on the whole cost function.

A. Trade-off between the server and network costs

In this subsection, we study the relationship between the network costs and the number of replicas in the network. The client loads in the experiments of the 50-node network of the previous section were scaled down by a factor of 100. This gave simpler and clearer results, and did not affect the performance of the different heuristics for the same delivery protocols. However, the network costs of the HMSM scheme are affected by this scaling. Hence, to study the differences between the protocols, we implement the MG-GCC and MG-GDD heuristics, which are regarded as nearly optimal solutions with original client loads on the PR, Waxman, and TS networks. The client loads are set to $L^1_{C^1} = 1000$, $L^2_{C^2} = 100$, and $L^3_{C^3} = 10$. Furthermore, we assume that the start-up delay of the PB scheme is 5 minutes to make the bandwidth consumption closer to that of the HMSM scheme.

Define the cost function as

$$C = \alpha \cdot D_{\text{start-up}} + \beta \cdot B_{\text{network}} + \gamma \cdot N_{\text{server}}.$$

The term B_{network} is calculated based on the weights of links, which are used to represent the number of hops in each link. It assumes that all hops have equal costs per unit of network bandwidth. The quantity N_{server} is the number of replicas, and $D_{\text{start-up}}$ is the start-up delay ratio. These three factors have different units, so the cost function uses three weights α, β, and γ, to unify them as the delivery costs. The resulting terms $\alpha \cdot D_{\text{start-up}}$, $\beta \cdot B_{\text{network}}$, and $\gamma \cdot N_{\text{server}}$ are called the protocol costs, the network costs, and the server costs, respectively.

Figure 2 shows the changes in delivery costs with different numbers of replicas based on different values of γ in the PR and Waxman network. Figure 2(a) illustrates the situation in which the delivery costs are just the network costs. For both the HMSM and the PB protocol, the changes for random topologies (i.e., the PR and the Waxman network) are almost linear. Figure 2(b) and (c) indicate that $\gamma = 50$ is the value around which the delivery costs change from a monotone decreasing function to a monotone increasing one. When $\gamma \geq 100$, the delivery costs increases in the number of replicas, which is depicted in Figure 2(d). A similar behaviour also holds for TS networks. However, because the network costs incurred in an TS network are a lot higher, the corresponding values that make the curves change significantly are also larger.

B. Trade-off between the protocol and network costs

In the previous section, the experimental results for the PB scheme did not include additional costs caused by the start-up delays. However, since the client satisfaction of a service clearly depends on these start-up delays, it is natural to introduce additional protocol costs because of the differences generated by the delivery protocols. In practice, measurements will decide whether the networks costs saved by the PB scheme balances the added protocol costs due this delivery protocol. Based on the results in last subsection, we try to get derive some rules to deal with this trade-off in this subsection.

Figure 3 illustrates the changes in delivery costs for different start-up delays. We choose the PR, the Waxman, and the TS network with the corresponding number of replicas of 1, 5, and 10, respectively. The optimal start-up delay depends on both the concrete network topology and on the value of α. Figure 3(a) illustrates the changes in delivery costs for various start-up delays based on values of $\alpha = 2000$, 3000, and 4000. The figure shows that when $\alpha = 3000$, the PB scheme with a start-up delay of 30 minutes can achieve lower delivery costs than the HMSM scheme. For the values $\alpha = 2000$ and 3000, the break-even points for the start-up delays are at 20 and 40 minutes, respectively. A similar analysis can also be applied

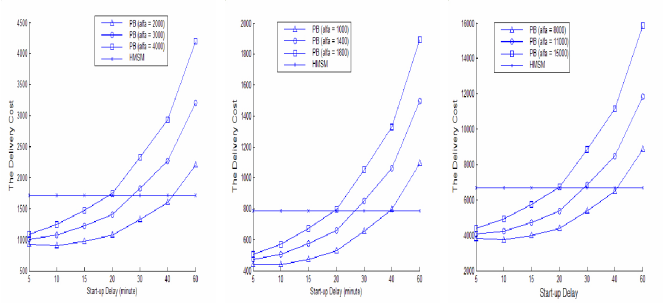

Fig. 3. Delivery costs vs. start up delays for the PR (1 replica), the Waxman (5 replicas), and the TS (10 replicas) network with $\beta = 1$ and $\gamma = 0$.

to Figures 3(b) and (c).

C. Hybrid strategies for multiple files

In streaming CDNs with multiple files, one needs to make a decision on whether one shares the replica placement, the delivery routing, and the related protocols. In the previous sections, multiple files shared the replica placement with potentially different routing (based on the same protocol). However, one could also use different replica placements and delivery routing. The benefit of this hybrid strategy becomes clear when one realizes that requests of different multimedia files are highly unbalanced. The top 10 files are requested in almost 90% of the cases, which can be modeled by Zipfian distributions [21].

In order to compare the two strategies, we use three different magnitudes of the client loads for the three files in our experiments. The load on file 1 is so large that it almost dominates the server placement and the routing. Under the first strategy, this would mean that the best placement and routing solution for the three files system is the same as that for the system with only file 1. But under the latter strategy this may not be the best solution for file 2 and file 3 separately. The PB protocols save network bandwidth but are associated with a start-up delay, whereas the HMSM protocols can support immediate services but consume more network bandwidth. Therefore, for the hybrid strategy, one could use a mixture of PB protocols to serve the multimedia files with high client loads, and HMSM protocols for files with low client loads.

Figure 4 show the results for $\alpha = 400$. The legend shows the protocols used for file 1, 2, and 3 in sequence. The start-up delay of the PB protocol is set to 20 minutes. Figure 4(a) shows that the hybrid strategy that uses the PB protocol for files with high loads only outperforms the single PB strategy. Recall the client loads are set to $L^1_{C^1} = 1000$, $L^2_{C^2} = 100$, and $L^3_{C^3} = 10$, so that it is obvious that the costs of using the PB scheme to deliver files 1 is more costly than that of the HMSM scheme. Figure 4(b) illustrates that the PB scheme is more expensive than using the HMSM scheme for files with low client loads.

VI. Conclusions

We have explored the design of streaming CDNs to attain (close to) minimum delivery costs. This is achieved by developing an optimization model to determine the optimal replica placement and the delivery trees for multiple media files. Our model can deal with different scalable streaming protocols, including periodic broadcast protocols and hierarchical multicast

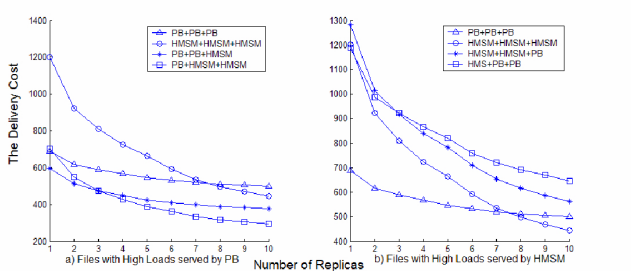

Fig. 4. Comparison between simple strategies and hybrid strategies.

streaming merging protocols. We introduced fast and accurate heuristic algorithms for solving the optimization model so that it can be applied to large streaming CDNs. Moreover, we discussed the trade-off between the start-up delay and the network bandwidth, and the trade-off between the number of replica servers and the network bandwidth. The results show that the heuristics are well-suited to be used in the design and optimization of commercial streaming CDN infrastructures.

References

[1] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman & Co., 1979.
[2] B. Li, M. J. Golin, G. F. Italiano, X. Deng, , and K. Sohraby, "On the optimal placement of Web proxies in the Internet," in *Proceedings of IEEE INFOCOM '99*, New York, NY, USA, 1999, pp. 1282–1290.
[3] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," *IEEE/ACM Trans. on Networking*, vol. 8, no. 5, pp. 568–582, 2000.
[4] S. Jamin, C. Jin, A. R. Kurc, D. Raz, and Y. Shavitt, "Constrained mirror placement on the Internet," in *INFOCOM*, 2001, pp. 31–40.
[5] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, "On the placement of Web server replicas," in *INFOCOM*, 2001, pp. 1587–1596.
[6] M. Karlsson, C. Karamanolis, and M. Mahalingam, "A framework for evaluating replica placement algorithms," HP Labs, Tech. Rep. HPL-2002-219, 2002.
[7] P. Radoslavov, R. Govindan, and D. Estrin, "Topology-informed Internet replica placement," *Computer Communications*, vol. 25, no. 4, pp. 394–402, 2002.
[8] P. Crescenzi and V. Kann, "A compendium of NP optimization problems," 2005.
[9] Y. Zhao, D. Eager, and M. Vernon, "Network bandwidth requirements for scalable on-demand streaming," 2002.
[10] Z. Fei, E. Zegura, and M. Ammar, "Multicast server selection: problems, complexity and solutions," *IEEE Journal on Selected Areas in Communications*, vol. 20, 2002.
[11] E. Brosh and Y. Shavitt, "Approximation and heuristic algorithms for minimum delay application-layer multicast trees," in *Proc. of the 23th Conference of the IEEE Computer and Communications Societies*, 2004.
[12] J. Almeida, "Streaming content distribution networks with minimum delivery cost," Ph.D. dissertation, University of Wisconsin, 2003.
[13] A. Hu, "Video-on-demand broadcasting protocols: a comprehensive study," in *INFOCOM*, 2001, pp. 508–517.
[14] D. Eager, M. Vernon, and J. Zahorjan, "Minimizing bandwidth requirements for on-demand data delivery," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 5, pp. 742–757, 2001.
[15] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions of Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.
[16] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *IEEE INFOCOM*, vol. 2, 1996, pp. 594–602.
[17] GT-ITM, "GT internetwork topology models," 2004. [Online]. Available: http://www.cc.gatech.edu/fac/Ellen_Zegura/gt-itm
[18] B. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, 1988.
[19] H. W. Holbrook and D. R. Cheriton, "IP multicast channels: Express support for large-scale single-source applications," in *ACM SIGCOMM '99*, New York, NY, USA, 1999, pp. 65–78.
[20] Y.-H. Chu, S. G. Rao, and H. Zhang, "A case for end system multicast," in *ACM SIGMETRICS '00*, New York, NY, USA, 2000, pp. 1–12.
[21] G. Zipf, *Human Behavior and the Principal of Least-Effort*. Cambridge, MA: Addison-Wesley, 1949.