Modeling the Location Selection of Mirror Servers in Content Delivery Networks

Peter Hillmann, Tobias Uhlig, Gabi Dreo Rodosek, and Oliver Rose
Universität der Bundeswehr München
Neubiberg, 85577, Germany
Email: {peter.hillmann, tobias.uhlig, gabi.dreo, oliver.rose}@unibw.de

***Abstract*—For a provider of a Content Delivery Network (CDN), the location selection of mirror servers is a complex optimization problem. Generally, the objective is to place the nodes centralized such that all customers have convenient access to the service according to their demands. It is an instance of the k-center problem, which is proven to be NP-hard. Determining reasonable server locations directly influences run time effects and future service costs. We model, simulate, and optimize the properties of a content delivery network. Specifically, considering the server locations in a network infrastructure with prioritized customers and weighted connections. A simulation model for the servers is necessary to analyze the caching behavior in accordance to the targeted customer requests. We analyze the problem and compare different optimization strategies. For our simulation, we employ various realistic scenarios and evaluate several performance indicators. Our new optimization approach shows a significant improvement. The presented results are generally applicable to other domains with k-center problems, e.g., the placement of military bases, the planning and placement of facility locations, or data mining.**

***Keywords*-CDN-ISP collaboration; k-center; profile correlation;**

I. INTRODUCTION

Content Delivery Networks (CDN) are large distributed systems of servers, which provide a caching infrastructure. To face the rapidly growing demand of content, CDN providers offer large data capacity within distributed storage infrastructures across the Internet. It is an efficient and common method to provide content to end-users providing high availability and high performance. The mirror servers copy and cache the content of an original source. Locations of mirror servers have to be determined with a short distance to customers with a view to provide adequate service and convenient access. Customers request content from a website and the according answer should be returned by the server within the shortest distance to speed up the responsiveness. Usually, the implementation is transparent to the user. This improves the Quality-of-Experience (QoE) for a customer as well as reducing transfer time to requests and load times for content. To ensure this, the corresponding content has to be available on the caching server. In most cases, the stored content is from contracting companies and offloaded content of Internet service providers (ISP). Nevertheless, a single server can not provide all the requested content immediately. If content is not available, then it is fetched from another mirror server or the original source. Furthermore, a cache replacement strategy has to decide which content is stored on the server. Large CDN providers like Akamai, Limelight, and Level 3 Communications can handle enormous amounts of requests. Beside an improved service quality for the customer, advantages are decreased network load and thereby reduced transmission costs for an ISP.

To that end, a predefined amount of mirror servers has to be placed and connected strategically to the ISP network infrastructure. An example of the problem is shown in Figure 1. The left side presents the geographical location of the network nodes and the edges of an infrastructure [1]. The right side includes the additionally registered mirror servers. The blue lines represent the assignments of the network nodes to their closest server and thereby the intended customer allocation.

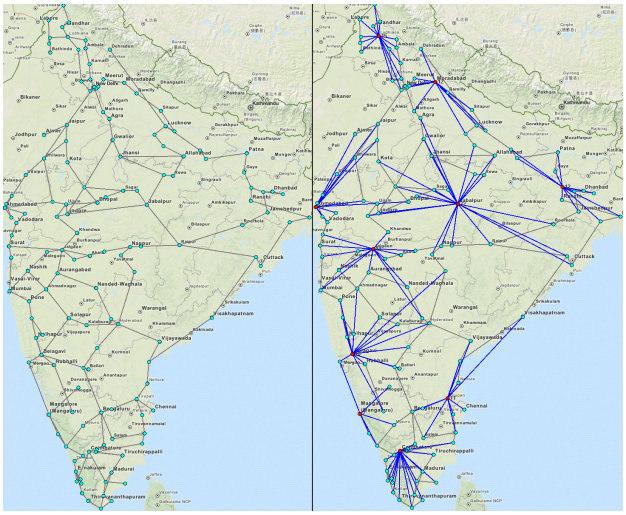

Figure 1. Example of a scenario for the placement of mirror servers in India.

In this work, we focus on the geographical and infrastructural placement of the servers enforcing the placement close to an already existing network node. We present a model of the problem and analyze the influence of different placement decisions. We simulate and automatically optimize the placement of the mirror servers and the assignment of customers to their most appropriate server with fitting content. The assignment has a strong influence on the Cache-Miss-Ratio as well as access latency and network load. To this end, we use simulation based optimization to analyze the necessary amount of mirror servers for an effective supply with respect to customer demands. Preliminary results on the caching behavior are discussed. Our results support the management in its difficult decision process during a planning phase. The

underlying problem is of importance in other application areas. For example in logistics, the placement of warehouses, fire stations and hospitals, as well as other facilities are based on the similar k-center problem.

Our paper has the following structure. After the introduction in Section I, we describe a typical scenario of our application area in Section II. We briefly discuss the complexity of the problem and introduce related optimization techniques in the Sections III and IV. The description of our model and the novel optimization algorithm are explained in Section V. The evaluation and assessment are presented in Section VI. The reference algorithms are outlined in Section VII. We summarize the paper and discuss future work in Section VIII.

II. SCENARIO

Consider the following scenario: a CDN provider intends to expand its business and wants to set up a completely new storage infrastructure in a country. The first step of the planning process is to analyze the network infrastructure and the connected users. Figure 2 shows an abstract example of a typical infrastructure which is non-hierarchical. To reduce complexity, multiple customers connected to the same gateway or network node are merged to a single node. Different approaches to generate the map are described in Section IV.

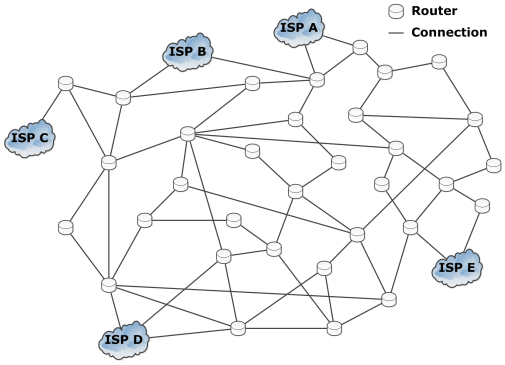

Figure 2. Abstract view of a network infrastructure from an ISP.

The CDN provider tries to place the mirror servers close to the customers. In this case, „close“ refers to the network infrastructure and not to the geographical location. The management has to decide where to place the mirror servers geographically as well as structurally and organize the assignment of the customers to a server. Furthermore, the distances have to be balanced against the correlated customer demands. The various customer demands are summarized in a profile. Customers send requests for specific content according to their profile to the server. This influences the dynamic caching behavior and the currently available content on the server. The requested content on a server should be similar to increase the cache hit ratio. This lowers the amount of requests from a mirror server to the original source, lowers the network utilization and reduces access latency. For the management of a CDN provider, it is important to know the necessary quantity of mirror servers for a given scenario to achieve a predefined objective. An example of such an optimization problem is shown in Figure 1.

Due to the complexity of the challenge, the placement of mirror servers and the assignment of customers to a server have to be technically supported with simulation and optimization. According to the scenario and various application areas, we need to answer the following important questions:

- How to model the system capturing its distinct properties, e.g. customer priorities and connection bandwidth?
- Where are mirror servers to be placed in the infrastructure in order to obtain short and efficient transmission paths?
- Which customer is assigned to which mirror server?
- How large is the influence of the positively correlated customer demands for a dedicated mirror server on the caching behavior?

III. PROBLEM DESCRIPTION

The scenario described in Section II is based on the k-center problem. For a given set of locations V, a predefined number of central locations K have to be determined. The k-center problem considers the minimization of the maximum distance between a location and its nearest center. In our case, it is the maximum distance of a customer to its assigned mirror server. The problem is proven to be NP-hard [2]. The problem can be specified using a strongly connected graph topology $G(V,E)$ with vertices ($v_i \in V \; with \; i := \{1,..,n\}$) and edges ($e_p \in E \; with \; p := \{1,..,m\}$). The vertices have different priorities and the edges have different weight depending on their characteristics. The k-center problem is defined on a complete, undirected graph. The objective function d defines the fitness value d(v_i,v_j) for an edge e(v_i,v_j) between two vertices (v_i,v_j), satisfying the triangle inequality. It selects the best edge from a vertex v_i to one of the calculated locations of a center node ($k_u \in K \; with \; u := \{1,..,l\} \; and \; l \leq n$). A center node k_u can only be placed on a location of a vertex. Considering our domain, a mirror server has to be connected to an existing network node. The set of all fitness values d from every vertex v_i is defined as D. We intend to place the number of center nodes K to minimize the maximum $d(v_i, k_u)$ from a vertex v_i to its best k_u. The objective criterion is calculated using Equation 1:

$$D_{center}(K) = min \max_{v_i=1,...,n} \min_{k_u \in K} d(v_i,k_u) \qquad (1)$$

IV. RELATED WORK

There is a lot of existing work on geographical mirror server placement. Most research focuses on special hierarchical topologies. A line structure and a tree structure are analyzed

in [3] and [4]. The work of [5] and [6] propose solutions for the placement of storage servers in CDN with tree-like topologies. An approach for a ring-based architecture is described in [7]. However, these topologies do not reflect realistic infrastructures and the results of the specific cases cannot be adapted to the more general case. There is less information about the placement of servers within an non-hierarchical or non-specialized infrastructure. Jamin et al. [8] analyzed different graph theoretic algorithms to determine the number and the placement of boxes for the purpose of network measurement with min k-center [9] and k-HST [10]. But these works have little relation to the area of CDN. Rana and Garg [11] proposes multiple heuristic approaches. An important and more general work is from Jamin et al. [12]. We use their proposed algorithms as benchmarks. However, either the aforementioned work covers our requirements only partially or the proposed approaches show only a modest performance. Furthermore, these publications do not provide an adequate modeling of the problem. They use rough models without priorities and weighting and pay no attention to customer demands. Furthermore, these publications do not answer all the questions posed in Section II.

Before the servers can be placed, the network infrastructure and the location of the customers need to be determined. Generally, the ISPs know the location of their customers and network nodes. If this information is not available one can obtain the information in an automated fashion. In [13] an IP geolocation service is used to gather user locations in terms of latitude and longitude. Users are clustered based on their coordinates to build the network nodes. We assume that a corresponding scenario is given and we use the official data of *the Internet Topology Zoo* [1].

V. Modeling

For the creation of an adapted model, the field of application is analyzed in more detail with specific characteristics. The entire system is strongly dependent on the underlying infrastructure. For an effective supply with the provided service, mirror servers should be connected strategically to the network infrastructure. Due to the typical expected high amount of requests from widely distributed customers, the servers need a high-performance connection with transit possibility. This is reached through a direct connection of the planned server to the backbone network in the TIER 1 or 2 topology. These top-level infrastructures provide a high-performance accessibility, which makes them an ideal choice. Therefore, we focus on mirror servers placed within the backbone network topology. For our model, a server can only be connected to an existing network node. This corresponds to the placement of the server at this node. A placement of a server on a data link or in an area without direct connection option is seen as impractical.

With regard to customer modelling, our approach uses the input information from the underlying scenario. It includes the geographical location of the customers and their connection to the Internet gateway as well as the infrastructure. To reduce the high amount of data for the modeling, we group multiple customers to a single network node. A group of customers connected to the same access node is aggregated with combined properties. This abstraction level is precise enough, because every Internet request from a customer has to pass this network node. Furthermore, the demands of the group of customers are aggregated to a group profile. To evaluate different groups of customers, the network node is assigned a priority. It is rated dependent on the represented amount of customers, their importance for the CDN provider, their payment, and frequency of requests.

Since we do not have real data of customer profiles from ISPs or CDN providers, pseudo realistic data is generated using the established model of Zipf-distribution [14]. To simulate the caching behavior each mirror server has been provided with a predefined amount of cache. The employed cache replacement strategy is *Least Frequently Used (LFU)*. During simulation, every group of customers sends repeatedly requests to their assigned server. Three different possibilities can occur during a repetition: no request, a request following the profile, or a novel request not according to the profile is send.

These aspects result in an abstract model of our scenario, see Figure 3. This includes prioritized network nodes for groups of customers and for placement possibilities of mirror servers. These network nodes are connected with each other via several data links using the existing infrastructure. The requested content is described by the profile of the network node, which is symbolized with a pie chart.

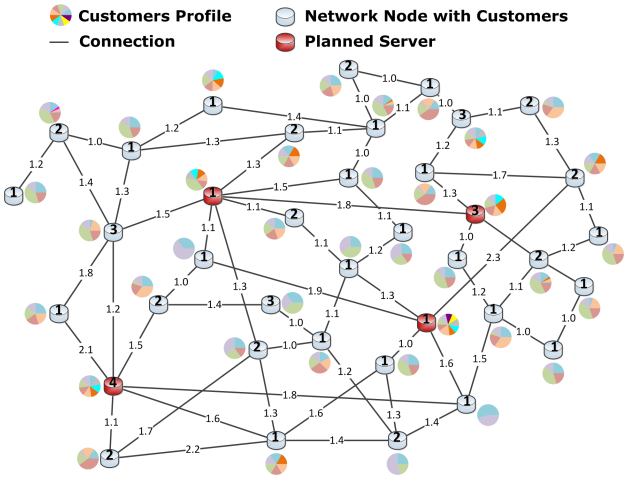

Figure 3. Example of the abstract model with a non-hierarchical network infrastructure including weighted edges and grouped customers to a prioritized network node.

For the optimization and the calculation of favorable server locations, it is mandatory to evaluate multiple locations with regard to the connection possibilities of assigned customers. To calculate the distances between the network nodes,

we use the metric of hops combined with the connection properties. A hop is a routing network component which is passed by a data packet during the communication. The delay of a transmission is mainly influenced by the processing of the network nodes themselves rather than by the physical length of a connection and the specific transmission medium. Therefor, all connections between the network nodes have a uniform virtual length of 1. The calculated distance between two nodes in the infrastructure results directly from the hop count. For the distance function and path finding method, we use Dijkstra's algorithm.

Nevertheless, we have to pay attention to the specific properties of a data link to model the infrastructure more precisely. To compare different, shortest paths from a customer to several servers, a fitness value is necessary. Figure 4 illustrates the problem. We assign to every single edge in our model a specific weight, dependent on maximum bandwidth, average utilization, and mean delay. The necessary information is provided by the ISP. We map and combine this weight of an edge with the hop to a virtual length larger than 1. A worse link quality leads to a longer virtual length. A high-performance connection has a virtual length close to 1, which represents the best value in our model. The edge weight transformation to a virtual length still enables us to use Dijkstra's algorithm, since it calculates the shortest path based on distances.

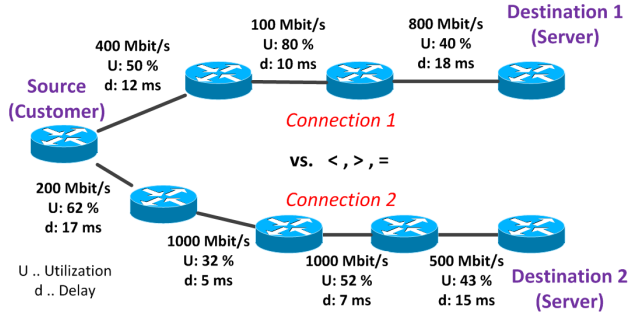

Figure 4. Evaluation problem of different connections from a customer to several servers.

The accumulated virtual path length from a customer to a server allows the comparison between different paths and enables efficient server assignments. To take the priority of a customer into account for the evaluation, the value of the path length is multiplied with the priority of the selected customer. Dependent on the selected optimization criterion, we minimize the maximum distance value of a customer, the average distance value of all customers or other objectives.

Figure 5 presents the model and the process of optimization. The data of the scenario is combined with the placement conditions to create an abstract model. The scenario contains the network infrastructure, locations of customers and their demands. The placement constraints include information on the specified amount of servers and the possible locations. Afterwards, the abstract model is simulated and the server locations are optimized. This is done with respect to the optimization criteria. It can be flexibly chosen to reflect the requirements of the management. In our case, we focus on a reduced network load with short transmission paths and high cache hit ratios. The process of optimization and simulation is iteratively repeated according to the used optimization algorithm. The result includes amongst other things the optimized server locations and the assignment of customers to their preferable server.

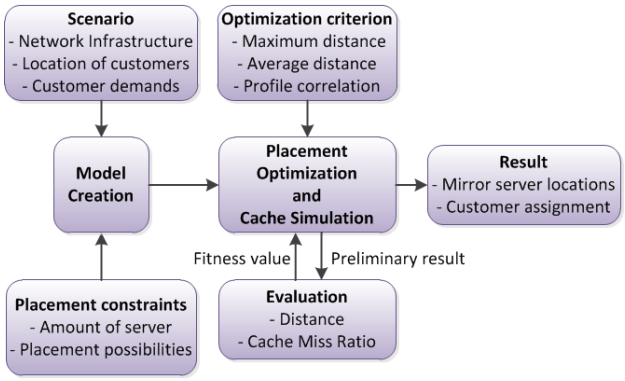

Figure 5. Modeling approach to optimize the placement of mirror server.

Figure 6 shows the generated model and the preferred mirror server locations. It is optimized with our developed algorithm *Dragoon* for shortest distance. The correlation of customer profiles is initially not considered or visualized, because the primary objective is distance.

Optimization Algorithm: Dragoon

We developed a new optimization algorithm Dragoon for server locations. The reference algorithms are described in Section VI. To reduce the effect of the sensitivity to initially randomly selected server locations and to avoid multiple runs, we create a deterministic initialization and optimization algorithm. Usually, the first placed server serves a high amount of customers and has a large impact on the resulting structure. This can lead to impractical configurations if we consider limited capacities. In these cases, an even distribution would be desirable to balance the load of servers. Dragoon consists of two main steps. After the initialization, the vertices are assigned to the nearest server location. In an iterative optimization these locations are improved.

In the preliminary stage of the initialization phase, an orientation mark is placed at the optimal location according to the one server placement problem. The first server is placed at the position of a network node which is farthest away from the orientation mark depending on network distance . This mark is only for orientation to place the first server and is removed after the first location was identified. Subsequently, the remaining amount of server is planned using the adapted Two-Approx strategy. It calculates for every network node the distance to all servers with Dijkstra's algorithm. It chooses the location with the largest distance to its closest server as the additional server location. After the

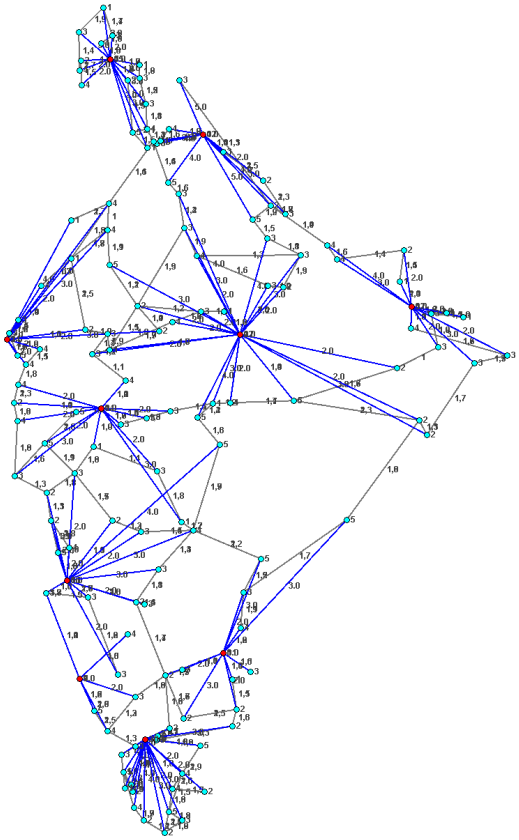

Figure 6. Example model of the scenario in India and the planned mirror server including the assignment of requests.

initialization, the algorithm starts with the iterative refinement to recalculate and further optimize the server locations. These newly designed optimization steps are adaptable to different placement constraints.

The algorithm tests all possibilities of better server locations among all network nodes within one hop distance and direct connection from the current location in each iteration step. The new location is chosen according to the best improvement. The optimization steps are iteratively calculated for each server. The improved location is evaluated with respect to the entire scenario, because a single change influences the complete system. In every iteration step, the customers are reassigned to their nearest server. All actual server locations are used in every evaluation step except of the observed one. This is done with respect to the specified optimization criterion, the maximum distance calculated by Dijkstra's algorithm.

If this main fitness value is unchanged, the algorithm will use another additional criterion. To choose between two solutions and to identify an improvement, we use an average or mean criterion. In each iteration, every server is allowed to shift its location only once. This leads to a stepwise improvement and avoids a premature stagnation in a local optimum. The order of the server selection has no significant influence on the final result. This is due to the global view onto the problem. For our simulations, the servers are chosen with respect to the largest distance of assigned user first. This iterative optimization is repeated until all server locations do not change any more. Due to the described initialization, only a few iterations are necessary until the algorithm terminates.

The algorithm accepts only improved locations in every iteration step. Therefore, the 2-approximable condition of algorithm Two-Approx holds. Nevertheless, we show that the performance is much better, close to the global optimum. This optimization is calculated in polynomial runtime and it will always terminate. The algorithm can also be adapted to upgrade an existing scenario with partly fixed servers from the beginning or other constraints. A typical application area for this algorithm is the clustering of data.

VI. REFERENCE PLACEMENT ALGORITHMS

The following algorithms are used as reference for our own development for the optimized placement without giving attention to customer profiles. These are specifically designed for the k-center problem, except for the evolutionary algorithms. All approaches can also be adapted to upgrade an existing scenario with partly fixed centers.

- Integer Linear programming
- Two-Approx
- Greedy
- k-Means (MacQueen, k-Means++)
- Evolutionary algorithm (SEREIN Framework)

A. Integer Linear Programming

The problem is defined with mathematical equations and can be solved with integer linear programming (ILP). A solution is calculated with support of ILP-Solver, for example GUROBI or CPLEX. Both use an individual implementation of the Dual Simplex algorithm to obtain a solution in polynomial runtime even a totally unimodular matrix is given. The Simplex algorithm does not necessary calculate the global optimum due to internal model relaxation, transformation, and rounding. The following ILP model is in accordance with a simple non capacitive problem, which has an integrality gap of 2 [15]. This guarantees a solution with fitness $d \leq 2 \cdot optimum$.

- The node u_j is selected as location of a mirror server (1) or not (0), Equation 2.
- The number of selected nodes correspond to the specified amount k of mirror server, Equation 3.
- A connection between two nodes x_{ij} can exist (1) or not (0). This corresponds to the assignment of a customer to a defined mirror server, Equation 4.
- Each node must have a connection to exactly one mirror server, Equation 5.
- A customer can only be provided by a node, if this is selected as mirror server c_i, Equation 6.

- Fitness function: The objective is the minimization of the largest distance. (dstc(ij) is the distance between two nodes) , Equation 7.
- Objective criterion, Equation 8.

$$u_j \in 0,1 \,\forall j \quad (2) \qquad \sum_0^m u_j = k \quad (3)$$

$$x_{ij} \in 0,1 \,\forall j \quad (4) \qquad \sum_0^n x_{ij} = 1 \,\forall j \quad (5)$$

$$x_{ij} \leq u_j \,\forall i,j \quad (6) \qquad dstc(ij) * x_{ij} \leq z \,\forall i,j \quad (7)$$

$$z \rightarrow min \quad (8)$$

B. Two-Approx

Two-Approx guarantees a 2-approximable solution with cost d where $d \leq 2 \cdot optimum$. Therefore the maximum value of a distance from a customer to his nearest mirror server is not larger than twice the maximum considering the optimal placement of the mirror servers [16]. This guarantee is given without the knowledge of the optimum value. This algorithm is the best approximation we can get in polynomial run time and it is well studied [17]. With these results, we are able to estimate the global optimum.

At the beginning, the algorithm choose a random node, which becomes the location of the first server. After that, it calculates for every node the distance to all other nodes. It chooses the node with the largest distance to their closest server as new location of the next server. This routine is repeated until the specified quantity of mirror servers are reached.

C. Greedy

The Greedy strategy [12] iteratively places the mirror servers on the topology at the customer locations. During each iteration, it tries all different placement possibilities for the next server and ultimately selects the node that provides the biggest benefit with respect to the optimization criterion. The Greedy strategy repeats this process until all mirror servers are placed.

D. k-Means

The main idea of the following clustering algorithms is to define k center points, one for each cluster. In our case a cluster is a group of nodes. The algorithms place all center nodes at the same time. Various approaches exist for a fixed number of clusters, they differ mainly with regard to the initial placement of center points. The MacQueen [18] algorithm is one of the simplest k-Means algorithms. It relies on randomly selected locations, which are used as the initial locations of the mirror servers. Another typical initialization is used in the k-Means++ algorithm [19]. Here, the location of the first server is chosen randomly at a node. The other servers are also placed randomly at a location of a node, however the probability is skewed to favor certain locations. The selection probability for locations is increased proportionally with their squared distances to already selected locations.

After the initialization, the customers are assigned to their respective server. For each group of customers related to a shared server an updated server location is calculated. The positions of the server nodes are mapped to the nearest node either in every step or at the end of the optimization. This process is repeated until server locations do not change any more. After every iteration step, the customers are reassigned to the nearest mirror server. The algorithm is sensitive to the initially randomly selected mirror server locations and does not necessarily find the optimal solution. To reduce this effect, the algorithms are run multiple times.

E. Evolutionary Algorithms

The SEREIN framework [20] is used to implement the evolutionary algorithms. We employ the standard implementation of a genetic algorithm (GA) provided by SEREIN and use a population with 25 individuals evolving over 80 generations. In addition, a particle swarm optimization (PSO) and simulated annealing (SA) approach are implemented as well. The parameters for the algorithms were determined experimentally using meta-optimization.

VII. SIMULATION AND ASSESSMENT

For the simulation of the model, the experiments, and the entire system we use a prototypic implementation in Java. For the evaluation, we run multiple simulations and compare different optimization algorithms. We used classical geo-coordinates in the 2-dimensional space and the described hop distance combined with euclidean distance as metric. For the evaluation, the fitness function calculates the distance parameters: maximum, quantiles, median and average. To generate significant results, we performed repeated optimizations using five different network topologies from *the Internet topology zoo* [1]. All used real-world scenarios have a similar amount of backbone network nodes, approximately 100. The presented results are the average values of all scenarios for each algorithm.

Initially, the performance of our developed Dragoon algorithm is compared with the approaches of Two-Approx and ILP. Since the solutions of these reference algorithms have a proven performance, the results serve as basic benchmark. The solutions of Two-Approx vary because of the random initialization. Nevertheless, the 2-approximable condition is valid every run. Based on the results s of the Two-Approx and ILP we defined a theoretical limit for the optimum.

$$s \leq 2 \cdot Optimum_{real} \implies \frac{1}{2} \cdot s = Optimum_{theoretical} \quad (9)$$

Since the 2-approximable condition is always valid, the highest values of the multiple test runs can be used for the estimation of the theoretical optimum instead of the average

values. Figure 7 shows that Dragoon returns much better results than the other approaches. The Dragoon algorithm reaches a performance close to the estimated theoretical optimum. For larger server amounts, the quality of solutions and the improvement with additional servers stagnates for all algorithms. It indicates that we are already close to the optimum.

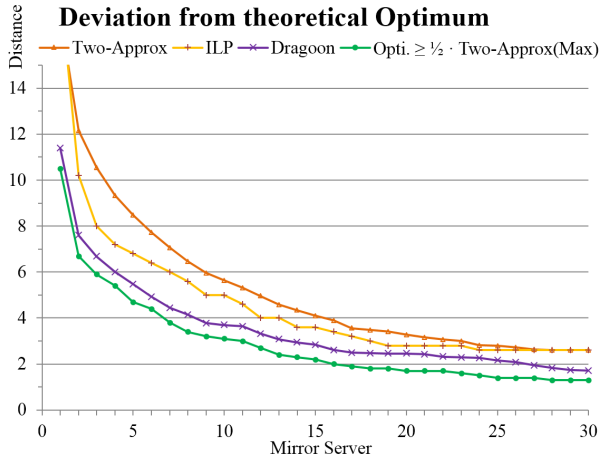

Figure 7. Comparison of Dragoon, Two-Approx, and ILP with the theoretical Optimum based on maximum distances.

Based on the maximum distance of a customer to its nearest mirror server, Table I and Figure 8 present the general Performance of the different algorithms. It shows that it is sufficient to set up a server at about 15 % of the nodes. After we placed 17 mirror server for about 110 network nodes, the average improvement of maximum distance for an additional server is less than 1 % with the Dragoon algorithm in all scenarios. While additional servers have a positive effect, increasing the overall capacity and load balance, the added benefit decreases significantly for large server counts. For high ratios of servers in relation to customer count we observe a saturation effect. The performance of the SEREIN framework with evolutionary algorithms is remarkably. SEREIN is not customized to this problem but reaches a comparably good solution as the specifically designed algorithms. The parameters of the heuristic were adapted after as little as six small test runs. The calculation time of all algorithms depends strongly on their parameters and predefined time limits. The time complexity of the algorithms is considerably different, but all optimization runs finished after a couple of minutes or a few hours. For such a fundamental decision as the placement of mirror servers, the computation time is acceptable. Decisions have strong long term effects and directly influence future service quality and costs.

A mirror server should have all requested data in its cache to reply immediately. The server caches mainly reactive and loads missing data from other sources at request time. Therefor, the customer requested content on a server shall be positive correlated so that the mirror server obtain requests

Table I
PERFORMANCE OVERVIEW WITH OBJECTIVE MAXIMUM DISTANCE IN RELATION TO AMOUNT OF MIRROR SERVERS.

Server	**Dragoon**	Two-Approx	ILP	Greedy	Greedy BT	GRASP	k-means	Serien GA	Monte Carlo	*Opti.-theoretical*
1	**11,4**	17,2	18,4	11,4	11,4	11,4	11,4	12,6	14,9	10,5
2	**7,6**	12,2	10,2	11,2	8	8	7,7	8,8	10,5	6,7
3	**6,7**	10,5	8	8,4	7,4	7	7,4	8,2	9	5,9
5	**5,5**	8,5	6,8	7,6	6,2	6,4	6,2	6,8	7,8	4,7
10	**3,7**	5,6	5	5,8	4,2	4,8	5,1	5,4	6,3	3,1
15	**2,8**	4,1	3,6	5	3,8	4,4	4,3	4,8	5,1	2,2
20	**2,5**	3,3	2,8	4,2	3,2	3,8	3,9	4,2	4,5	1,7
30	**1,7**	2,6	2,6	4	2,8	3,2	2,9	3	3,8	1,3

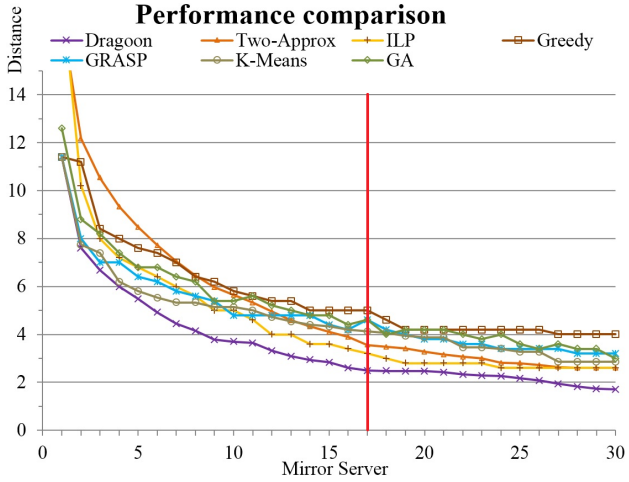

Figure 8. Performance comparison of different algorithms. The red vertical line marks the quantity of mirror server, after which the average performance improvement is less than 1 %.

for current available data. This increase the cache hit ratio on a server, lowers the network utilization and reduces access latency. Furthermore, it avoids redirecting to other servers and lowers the load on the original source. In the following Experiment we compare different optimization objectives. On the one hand, the optimization has been in terms of minimizing the distance between the customers and mirror servers, regardless of the customer profiles. On the other hand, it has been optimized with regard to a positive correlation of customer profiles at the mirror server. To optimize the profile correlation, we used the Spearman's rank correlation coefficient. Customers are reassigned after an initial placement optimization to the best fitting mirror server. Following, updated locations of the servers are calculated based on the assignment. Table II shows the comparison of the two conflicting objectives.

Table II
BENCHMARKING WITH RESPECT TO DIFFERENT OPTIMIZATION OBJECTIVES.

	Objective: Distance	Objective: Profile
Average request distance	1,6 Hops	6,1 Hops
Cache Miss Ratio	36 %	5 %

VIII. CONCLUSION AND OUTLOOK

In this paper, we propose an adaptive model for the placement problem of mirror server. We have developed a new placement strategy, which outperforms the others and finds solutions close to the optimum in short time. To enhance the QoE of customers and the access performance, we analyzed the quantity of recommended mirror servers for a predefined area. After a specific number of servers is placed for a CDN, it is more important to improve the cache hit ratio than to further reduce the distance and access latency with additional mirror servers. Our study showed a significant cache hit improvement by optimizing positive correlated customers profiles instead of just reduced network distances. The trade-off between these contradictory objectives has to be taken into account by the management of the CDN provider. For an effective and balanced interim solution much more effort is necessary. If we add more mirror server than 15% of the number of network nodes, we reached a saturation effect. Our analysis shows, even for the best placement strategy less than 1 % performance gain can be expected per additional server. This is important for management decisions between adding new mirror servers or extending existing ones. With improved performance of the placement algorithms, we can further decrease the number of server locations. In the future, we intend to further enhance the performance of the placement algorithms. Another open question is the optimal distance to the second closest server to provide a high fault tolerance. We are working on an optimization, which reaches good solutions for both objectives at one time, reduced distance and high cache hit ratio.

ACKNOWLEDGMENT

This work was partly funded by FLAMINGO, a Network of Excellence project (ICT-318488) supported by the European Commission under its Seventh Framework program.

LITERATUR

[1] S. Knight, H.X. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *Selected Areas in Communications*, October 2011.

[2] P. Potikas. The k-center problem - approximation algorithms. Spring, 2009.

[3] G. Carofiglio, M. Gallo, L. Muscariello, and D. Perino. Modeling data transfer in content-centric networking. Teletraffic Congress (ITC), September 2011.

[4] P. Krishnan, D. Raz, and Y. Shavitt. The cache location problem. Transactions on Networking, October 2000.

[5] M. Ionut Andreica and N. Tapus. Central placement of storage servers in tree-likeike content delivery networks. EUROCON, May 2009.

[6] G. Carofiglio, V. Gehlen, and D. Perino. Experimental evaluation of memory management in content-centric networking. International Conference on Communications (ICC), June 2011.

[7] Z. Wang, Hai Jiang, Yi Sun, Jun Li, Jing Liu, and E. Dutkiewicz. A k-coordinated decentralized replica placement algorithm for the ring-based cdn-p2p architecture. Symposium on Computers and Communications (ISCC), June 2010.

[8] S. Jamin, Cheng Jin, Yixin Jin, D. Raz, Y. Shavitt, and Lixia Zhang. On the placement of internet instrumentation. INFOCOM. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. (Volume:1), March 2000.

[9] Vijay V. Vazirani. Approximation algorithms. Springer-Verlag New York, 2001.

[10] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. Foundations of Computer Science, October 1996.

[11] R. Rana and D. Garg. Heuristic approaches for k-center problem. International Advance Computing Conference (IACC), March 2009.

[12] S. Jamin, Cheng Jin, A.R. Kurc, D. Raz, and Y. Shavitt. Constrained mirror placement on the internet. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), April 2001.

[13] S. J. Jafari and H. Naji. Geoip clustering: Solving replica server placement problem in content delivery networks by clustering users according to their physical locations. Information and Knowledge Technology, May 2013.

[14] L. Breslau, Pei Cao, Li Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: evidence and implications. In *Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, March 1999.

[15] A. Archer, R. Rajagopalan, and D. B. Shmoys. Lagrangian relaxation for the k-median problem: New insights and continuity properties. 2003.

[16] T. Gonzalez. Clustering to minimize the maximum inter-cluster distance. Theoretical Computer Science 38, Elsevier Science B.V., 1985.

[17] D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. INFORMS: Mathematics of Operations Research, May 1985.

[18] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, 1967.

[19] D. Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. Eighteenth annual symposium on Discrete algorithms (SODA), 2007.

[20] T. Uhlig. Serein - a framework to model metaheuristics, 2015. URL `http://sourceforge.net/projects/serein/`.