On Communication Cost vs. Load Balancing in Content Delivery Networks

M. Jafari Siavoshani*, S. P. Shariatpanahi†, H. Ghasemi*, A. Pourmiri‡
*Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
†School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
‡Department of Software Engineering, Faculty of Computer Engineering, University of Isfahan, Isfahan, Iran
mjafari@sharif.edu, pooya@ipm.ir, hghasemi@ce.sharif.edu, a.pourmiri@comp.ui.ac.ir

***Abstract*—It is well known that load balancing and low delivery communication cost are two critical issues in mapping requests to servers in Content Delivery Networks (CDNs). However, the trade-off between these two performance metrics has not been yet quantitatively investigated in designing efficient request mapping schemes. In this work, we formalize this trade-off through a stochastic optimization problem. While the solutions to the problem in the extreme cases of *minimum communication cost* and *optimum load balancing* can be derived in closed form, finding the general solution is hard to derive. Thus we propose three heuristic mapping schemes and compare the trade-off performance of them through extensive simulations.**

Our simulation results show that at the expense of high query cost, we can achieve a good trade-off curve. Moreover, by benefiting from the *power of multiple choices* phenomenon, we can achieve almost the same performance with much less query cost. Finally, we can handle requests with different delay requirements at the cost of degrading network performance.

***Index Terms*—Content delivery networks, distributed load balancing, power of multiple choices, query cost.**

I. Introduction

Content delivery networks are becoming an indispensable architecture design of modern communication networks. As mentioned by Cisco in [1], 80% of Internet traffic will be of multimedia nature by 2019 and by this time half of the Internet traffic will be handled by CDNs. Many practical solutions are already deployed to manage this growing demand such as Akamai [2], Azure, Amazon CloudFront, and Limelight [3]. In such networks, a number of servers (mainly deployed in large scale data centers) are distributed geographically which cache contents from original provider near end-users. Then, upon arrival of each user request, it will be served by an appropriate CDN server. This will reduce congestion at the original content provider.

An important design problem in CDNs is the assignment of each request to an appropriate CDN server [4]. Usually an appropriate server is interpreted as the nearest one which has cached the file [5], [6], [7]. This interpretation leads to low *communication cost* when delivering content to the user. However, there is another performance metric which should be considered in practice, namely, *load balancing*. In other words, a practical assignment scheme should not impose a large number of requests to a single CDN server. To this end, to assign each request, the CDN mapping algorithm should consider the current load of servers. Then it should prevent assigning a request to a busy server which may result in choosing a far server from the request. Thus, these two metrics, namely, proximity and load balancing may be in contention in practical scenarios [8].

In this paper, we first formalize the trade-off between communication cost and load balancing in a CDN as a stochastic optimization problem. The solution of this problem is an assignment strategy which will result in the optimum communication cost vs. load balancing trade-off. In the extreme cases of the minimum communication cost and optimal load balancing performance the solutions can be obtained in closed form. However, due to the complexity of obtaining the optimal solution in general, we propose three heuristic request assignment strategies which result in different trade-off curves.

The first proposed scheme manages the trade-off between communication cost and load balancing by probabilistically switching between the two above mentioned extreme cases. The switching probability is the algorithm parameter which determines the operational point of the trade-off. In the second suggested scheme for assigning each request we define a *desirability value* for each server that has cached the requested file. This desirability value takes into account both communication cost and current load of these servers. Then the request is assigned to a server with the minimum desirability value. Finally, the third scheme benefits from the notion of *power of multiple choices* [9] in balanced allocation literature to manage the trade-off. In this scheme for each request we find Δ servers with least communication costs for responding to that request. Then the request is assigned to the one with the minimum current load among these Δ servers.

It should be noted that although in this paper we focus on the trade-off between load balancing and communicaion cost of the proposed schemes, another important practical metric is the overhead of each scheme due to queue length queries for each assignment. Thus, in this paper, we also compare the schemes in terms of the query overhead imposed to the network.

The idea of exploiting caching servers to bring data near end-users is well known and has been used in commercial systems as well, [2], [3]. However, the technical challenges introduced by such framework is still the topic of many active research works. Load balancing in CDNs, as one of the most

978-1-5386-1629-1/17/$31.00 ©2017 IEEE

important issues, has been treated through various approaches. In this challenge the mapping algorithm which assigns content requests from users to CDN servers should make sure that no server becomes overloaded. Among load balancing proposals, distributed approaches, such as [10] [6], and [11] have attracted special attention due to their more practical nature. One promising distributed approach is *randomized load balancing* benefiting from the power of multiple choices [9], [12], and [13]. In this approach, for assigning each request to an appropriate server, first the current load of a few number of randomly selected servers are queried. Then, the request is assigned to the server with minimum load to balance out the load of network.

The focus of all above works is balancing out network load without considering its interaction with the communication cost of delivering the content. In this paper, for the first time, we formalize the optimization problem which captures the fundamental trade-off between load balancing performance and communication cost. Moreover, in contrast to previous works, our schemes are able to manage this trade-off properly.

The rest of the paper is organized as follows. In Section II, we first explain the network model and formally define our metrics. Then in Section III, we formalize the trade-off between load balancing and communication cost through a stochastic optimization problem. Afterwards, three heuristic schemes are proposed which capture this trade-off. In Section IV, we compare the performance of proposed scheme through extensive simulations. Finally, Section V concludes the paper.

II. Problem Definition

We consider a content delivery network whose goal is to deliver contents from a library of N files $\mathcal{W} = \{W_1, \dots, W_N\}$, each of F bits, to the users requesting files. The network consists of three major elements (see Fig. 1):

- L servers each capable of storing MF bits.
- K users requesting files in continuous time according to K Poisson point processes with parameter λ_i for user $i \in [1:K]$.
- A communication network for transferring files from servers to users. The cost of transferring a file of F bits from Server j to User i is $c_{i,j}$. For convenience, we define a cost matrix $\mathbf{C} = [c_{i,j}]_{i=1,j=1}^{i=K,j=L}$.

In each request, the probability of requesting file W_i is p_i. Moreover, we let $\mathcal{P} = \{p_1, \dots, p_N\}$ represent the file popularity profile. In this paper, we assume that the file popularity profile follows the Zipf distribution with parameter β. In Zipf distribution the request probability of the i-th popular file is inversely proportional to its rank as follows

$$p_i = \frac{1/i^\beta}{\sum_{j=1}^{K} 1/j^\beta}, \quad i = 1, \dots, K,$$

which has been confirmed to be the case in many practical applications [14], [15]. Note that the case $\beta = 0$ results in a uniform file popularity distribution.

We assume that the network operates in two phases, namely, *cache content placement*, and *content delivery*. In the placement phase each server caches M files. We assume $LM \geq N$ so that all files are stored in the aggregate memory of servers. The content of Server i's cache is denoted by $\mathcal{Z}_i \subseteq \{1, \dots, N\}$ where $|\mathcal{Z}_i| \leq M$. In this paper we assume that in the cache content placement phase, file W_i is cached with probability p_i, i.e., proportional cache content placement.

In the delivery phase, when a user requests file W_i, the request will be mapped to a server which has cached this file. Then, this request will be appended to the server's queue. Each server's queue follows a First-In First-Out (FIFO) strategy with a given service time Probability Distribution Function (PDF) (e.g., exponential or constant).

As cache content placement phase is during network low-peak hours, the only restriction in this phase is the cache size of each server. In other words the communication cost in this phase is negligible compared to that of content delivery phase.

Suppose large enough time T has passed in the delivery phase. We define

- n_i = the number of requests of user i during this time interval. This is a random variable with $\mathbb{E}[n_i] = T\lambda_i$.
- $\mathbf{t}_i = (t_{i,1}, \dots, t_{i,n_i})$, where $t_{i,j}$ is the arrival time of j'th request of user i.
- $\mathbf{d}_i = (d_{i,1}, \dots, d_{i,n_i})$, where $d_{i,j}$ is the file index of j'th request of user i. This is an i.i.d. sequence with the distribution $\mathcal{P}$.
- $\mathbf{q}(t) = (q_1(t), \dots, q_L(t))$, where $q_i(t)$ is the queue length of Server i at time t.

As indicated above, upon each request arrival at time t an online mapping algorithm maps the request from the requesting user to a server with assuming full knowledge of servers' cache contents, servers' queue status $\mathbf{q}(t)$, and the communication cost matrix $\mathbf{C}$. This mapping can be formally defined as follows.

Definition 1. The mapping scheme of the above system is a function Ψ defined as

$$\Psi\big((i,j), \mathbf{C}, \mathbf{q}(t_{i,j})\big) \mapsto \{1, \dots, L\}, \tag{1}$$

where (i,j) denotes the jth request of User i and $t_{i,j}$ shows the arrival time of such a request.

Moreover, we define

- $\mathbf{c}_i = \big(c_{i,S_{i,1}}, \dots, c_{i,S_{i,n_i}}\big)$, where $c_{i,S_{i,j}}$ denotes the communication cost of serving j'th request of User i. Here, $S_{i,j}$ is the server responsible for jth request of User i, which is determined by the mapping scheme as $S_{i,j} \triangleq \Psi\big((i,j), \mathbf{C}, \mathbf{q}(t_{i,j})\big)$.
- $\boldsymbol{\tau}_i = (\tau_{i,1}, \dots, \tau_{i,n_i})$, where $\tau_{i,j}$ is the time interval from appending j'th request of User i to $S_{i,j}$'s queue, until the request is served.

Then for evaluating the performance of each specific mapping scheme Ψ, we consider two metrics. Our first metric is

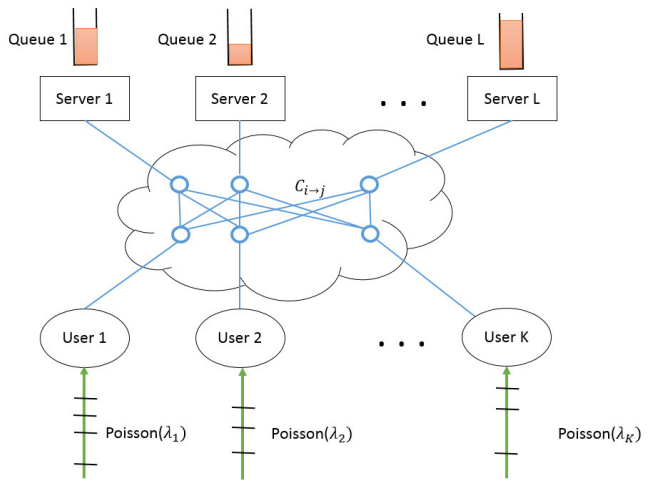

Fig. 1. A CDN with K users and L servers.

the *average cost* per file delivery

$$\bar{C}(\Psi) \triangleq \lim_{T\to\infty} \mathbb{E}\left[\frac{1}{\sum_{i=1}^{K} n_i} \sum_{i=1}^{K}\sum_{j=1}^{n_i} c_{i,S_{i,j}}\right]. \tag{2}$$

Note that the dependency of $\bar{C}$ to the forwarding scheme Ψ is through the random variables $S_{i,j}$.

Our second metric is the *average waiting time* for each request, defined as follows

$$\bar{D}(\Psi) \triangleq \lim_{T\to\infty} \mathbb{E}\left[\frac{1}{\sum_{i=1}^{K} n_i} \sum_{i=1}^{K}\sum_{j=1}^{n_i} \tau_{i,j}\right]. \tag{3}$$

Here $\bar{D}$ depends on the forwarding scheme Ψ through the length of servers' queues.

In the above definitions, the expectations are with respect to the randomness of servers service times, users arrival requests process, and any other randomness introduced by a random mapping rule.

Remark 1. The model introduced above is general-enough to cover various content delivery scenarios. As an example, the servers in the model can represent the *edge servers* of a CDN (say Akamai), which deliver content to costumers [2]. As another example, the servers and users can be the same devices in a device-to-device (D2D) setup, where each device has cached some files, and requests some other files [16].

III. Load Balancing vs. Communication Cost

As already mentioned in Section I, there exists a fundamental trade-off between average cost $\bar{C}$ and waiting time $\bar{D}$ in a distributed caching scenario. That is because when you look for low communication cost, upon each request arrival, you do not have much options from the servers pool to assign the request to. This will limit the load balancing power of the mapping algorithm. The above tension can be managed by the mapping rule used, i.e., Ψ. Formally, this trade-off is captured by the following optimization problem

$$\begin{aligned} &\min_{\Psi} \alpha\bar{C}(\Psi) + (1-\alpha)\bar{D}(\Psi) \\ &\text{s.t.} \\ &\quad d_{i,j} \in \mathcal{Z}_{\Psi\left((i,j),\mathbf{C},\mathbf{q}(t_{i,j})\right)}, \ \forall i \in [1:K], j \in [1:n_i], \end{aligned} \tag{4}$$

for arbitrary value of $\alpha \in [0,1]$. In the optimization problem (4), the minimization is over all mapping rules which have full knowledge of communication cost matrix $\mathbf{C}$, and instantaneous queue length status of all servers $\mathbf{q}(t)$. The parameter α determines the importance of each performance metric. Also the constraint ensures that the assigned server has cached the file in the cache content placement phase.

The solution to this optimization problem leads to the pareto-optimal trade-off curve between communication cost and waiting time. It should be noticed that the problem is hard to solve in the general case. Thus, in the following subsections, we propose three different algorithms as practical heuristic solutions to this optimization problem.

A. Scheme I: Probabilistic Scheme Switching

In order to gain more insight into the optimization problem (4), let us consider two special cases of $\alpha = 0$ and $\alpha = 1$. Setting $\alpha = 0$ in (4) puts the focus on minimizing the average waiting time. In particular, it can easily be observed that the optimal forwarding scheme Ψ in this case is

$$\Psi_1\left((i,j),\mathbf{C},\mathbf{q}(t_{i,j})\right) = \underset{k:\ d(i,j)\in\mathcal{Z}_k}{\arg\min}\ q_k(t_{i,j})$$

which means assigning each request to the server with minimum load that has cached the file, without considering communication costs (see for example [17]).

In contrast, for $\alpha = 1$ the focus is on minimizing average communication cost. In this case the optimal mapping scheme is

$$\Psi_2\left((i,j),\mathbf{C},\mathbf{q}(t_{i,j})\right) = \underset{k:\ d(i,j)\in\mathcal{Z}_k}{\arg\min}\ c_{i,k}$$

that means upon arrival of each request, it will be forwarded to the server with minimum cost, without considering current load of servers.

The above two special cases focus on minimum waiting time and minimum communication cost, respectively. A natural probabilistic generalization which considers both metrics is presented in Algorithm 1.

Algorithm 1 Probabilistic Scheme Switching (PSS)

Require: (i,j), ζ, $\mathbf{C}$, $\mathbf{q}(t_{i,j})$
1: $\Lambda \leftarrow \{k | k \in [1:L], d_{i,j} \in \mathcal{Z}_k\}$
2: Generate $x \in [0,1]$ uniformly at random
3: **if** $x \le \zeta$ **then**
4: Query $\{q_k(t_{i,j})\}_{k\in\Lambda}$
5: Assign the (i,j) request to $\Psi_1 = \arg\min_{k\in\Lambda} q_k(t_{i,j})$
6: **else**
7: Assign the (i,j) request to $\Psi_2 = \arg\min_{k\in\Lambda} c_{i,k}$
8: **end if**

In Algorithm 1, for each request, with probability ζ the minimum delay approach is used, and with probability $1-\zeta$ the minimum cost approach is used. This scheme mimics the situation where some requests have more stringent delivery delay requirements. Here, the ratio of such packets to total requests is determined by ζ. It should be noted that this algorithm needs on average $LM\zeta/N$ queue length queries for each assignment.

B. Scheme II: Weighted Metrics Combination

As discussed above the solution to the optimization problem in (4) for $\alpha = 0$ and $\alpha = 1$ can be obtained in a straightforward manner. However, if interested in both low communication cost and acceptable delay at the same time, we should consider other values of α as well. This naturally leads to the mapping scheme presented in Algorithm 2.

Algorithm 2 Weighted Metrics Combination (WMC)

Require: (i,j), α, $\mathbf{C}$, $\mathbf{q}(t_{i,j})$
1: $\Lambda \leftarrow \{k | k \in [1:L], d_{i,j} \in \mathcal{Z}_k\}$
2: Query $\{q_k(t_{i,j})\}_{k\in\Lambda}$
3: $\beta_1 \leftarrow \sum_{k\in\Lambda} c_{i,k}$
4: $\beta_2 \leftarrow \sum_{k\in\Lambda} q_k(t_{i,j})$
5: **for all** server $k \in \Lambda$ **do**
6: $\quad \eta(i) \leftarrow \alpha \frac{c_{i,k}}{\beta_1} + (1-\alpha)\frac{q_k(t_{i,j})}{\beta_2}$
7: **end for**
8: Assign the request to $\arg\min_{k\in\Lambda} \eta(k)$

In summary, in Algorithm 2, we assign the request to the server with minimum value $\eta(k)$ (called desirability value of user k), among those servers which have cached the request. For server k, the value of $\eta(k)$, is a weighted sum of communication cost and the current load of the server. The weight is determined by the parameter α. It should be noted that this algorithm requires on average LM/N queries for each assignment.

C. Scheme III: The Power of Multiple Choices

In Algorithm 3, we propose a scheme that reveals the fundamental trade-off between these two objectives via a different approach motivated by the *power of multiple choices* in the balanced allocations literature [9] (also, see [13]).

Algorithm 3 Multiple Choices Scheme (MCS)

Require: (i,j), Δ, $\mathbf{C}$, $\mathbf{q}(t_{i,j})$
1: $\Lambda \leftarrow \{k | k \in [1:L], d_{i,j} \in \mathcal{Z}_k\}$
2: Sort $\{c_{i,k}\}_{k\in\Lambda}$ and choose indices of the least Δ values: $k_1 \le \cdots \le k_\Delta$
3: Query $\{q_k(t_{i,j})\}_{k\in\{k_1,\ldots,k_\Delta\}}$
4: Assign the (i,j) request to $\Psi = \arg\min_{k\in\{k_1,\ldots,k_\Delta\}} q_k(t_{i,j})$

In Algorithm 3, upon arrival of each request the corresponding user first determines the set of servers which have cached the requested file i.e., Λ. By sorting these servers based on their communication costs to this user, Δ servers with lowest communication costs are determined. Then, this user compares the queue length of these Δ servers, and assigns the request to the least loaded one. Therefore, this algorithm needs Δ queries for each assignment.

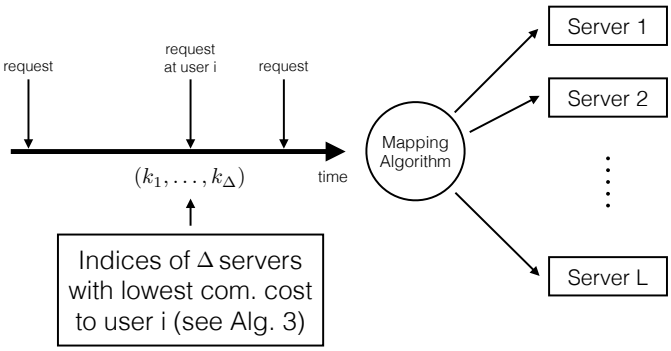

Fig. 2. Relation between Scheme III and the Supermarket Model studied in [12].

Algorithm 3 is motivated by the *Supermarket Model* investigated by Mitzenmacher et al. in [12]. In this model we have a number of servers serving a request arrival process. If each request is randomly assigned to a server, the system will face large queue delays. However, suppose each request first randomly chooses $\Delta \ge 2$ servers, and is then assigned to the least loaded one. It is shown in [12] that this will result in an exponential improvement in the average queue length of the servers.

In Algorithm 3, the metaphor of the results in [12] is used to reduce the queuing delay, at the expense of higher communication cost (see Fig. 2). Here, we have L servers serving requests from K users. Since each request arrives at a distinct user every request will choose Δ random servers with low communication cost, based on its requesting user cost vector. Thus, a vector containing indexes of Δ candidate servers can be considered to be attached to each request, where this vector depends on the index of the user, and the communication cost matrix $\mathbf{C}$. If we choose large values of Δ, then we have more options as our candidate servers to assign the request, which in turn reduces delay. However, this will introduce more communication cost, since servers with higher communication cost are let into the candidate set. Thus, Δ is the parameter managing the trade-off between delay and communication cost.

IV. Performance Evaluation

In this section, we investigate the performance of schemes proposed in Section III by developing an event-based simulation environment.

In our simulations, presented in the following, we assume that $L = K = 100$ and $N = 70$. The average incoming request rate for the ith user is $\lambda_i = 0.9$ and each servers' queue has an $\mathrm{Exp}(1)$ service process. Each simulation consists of 10^5 request events and every point on each graph is

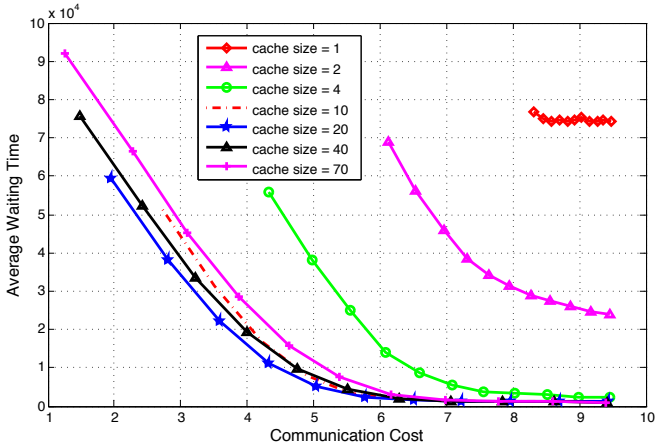

Fig. 3. Average waiting time, average cost trade-off for Algorithm 1 (PSS).

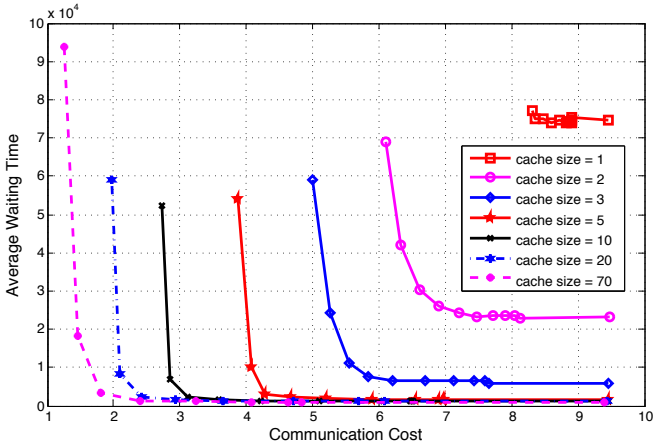

Fig. 4. Average waiting time, average cost trade-off for Algorithm 2 (WMC).

an average of 100 independent simulation runs. To generate the cost matrix $\mathbf{C}$, we consider that users and servers are distributed over a square Lattice uniformly at random. Then the communication cost between each user and a server is their Manhattan distance.

In the first set of simulations, the performance trade-off of Algorithm 1 is shown in Fig. 3 for different cache sizes. For each curve on the figure, the most left point corresponds to $\zeta = 0$, i.e., it leads to the minimum communication cost scheme. In contrast, the most right point corresponds to $\zeta = 1$ which results to the minimum average waiting time scheme. As ζ is varied between 0 and 1 the trade-off curve of Algorithm 1 is captured. From the figure we observe that for low cache sizes, i.e., $M \leq 4$, we have both high average waiting time and communication cost. In particular for $M = 1$, the average waiting time cannot be reduced even by introducing large communication cost.

Fig. 4 shows the performance trade-off of our second set of simulations. Here, the performance trade-off is captured by varying the parameter $\alpha \in [0, 1]$ in Algorithm 2. Interestingly, in contrast to Fig. 3, we can reduce the average waiting time by introducing just a small communication cost. Moreover, for the case $M = 1$ we observe the same behaviour as of Fig. 1.

The third set of simulations, investigates the performance of Algorithm 3 shown in Fig. 5. Here the parameter Δ manages the trade-off between the average waiting time and communication cost. The performance of Algorithm 3 trade-off management is similar to that of Algorithm 2; by introducing small communication cost the average waiting time is reduced significantly. Here, this phase transition can be explained by the well known power of two choices phenomenon discussed in [9], [12], [13].

In order to compare the performance of the three proposed schemes, their trade-off curves are plotted together in Fig. 6 for two values of cache sizes, i.e., $M = 2$ and $M = 70$. In general, Algorithm 2 has a slightly better performance than Algorithm 3 and both of them surpass Algorithm 1. It should be noted that, for large cache sizes, Algorithm 2 and 3

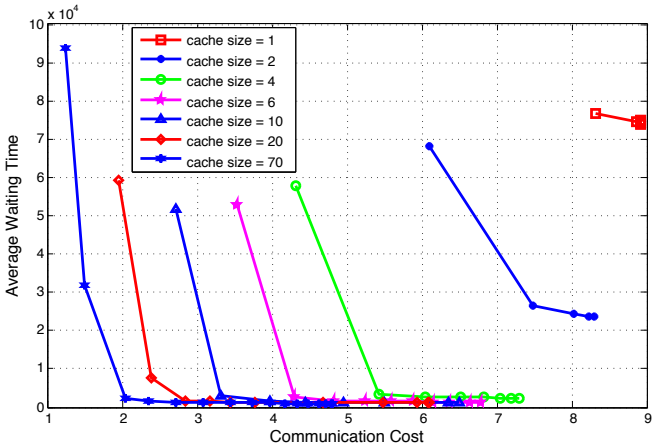

Fig. 5. Average waiting time, average cost trade-off for Algorithm 3 (MCS).

have almost the same performance. In addition in Fig. 7, the communication cost versus cache size for three schemes is plotted for a fixed average waiting time. This figure confirms the trend observed above for Fig. 6.

In summary, our simulation results show that Algorithm 2 has the best performance among all proposed schemes. In fact this algorithm solves an instantaneous optimization problem for each request assignment. Although this new optimization problem is different from our original problem (4) but they are closely related. This is why Algorithm 2 performs very well. However, the main drawback of this algorithm is that upon each assignment, it needs to query all the nodes that have cached the requested file. Interestingly, Algorithm 3 addresses this drawback with negligible performance loss. This is due to power of two choices which is a well investigated phenomenon in load balancing literature. Finally, though Algorithm 1 has the worst performance, it has the capability of handling the requests with different delay requirements. Table I summarizes query costs of each algorithm.

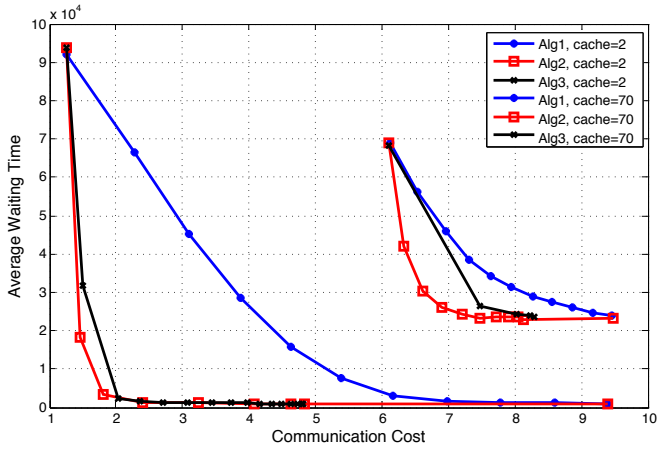

Fig. 6. Comparison of proposed algorithms for two different cache sizes.

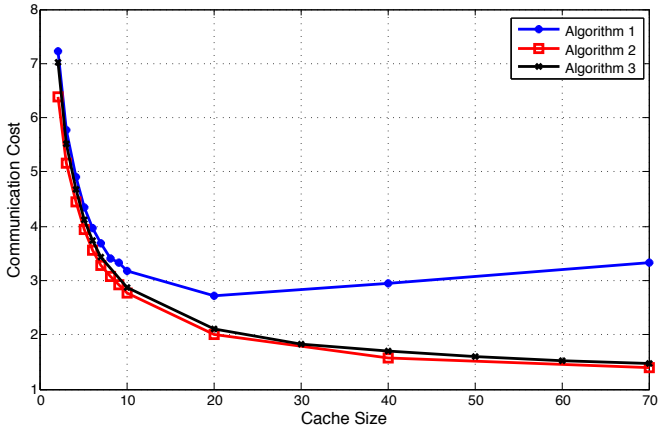

Fig. 7. Communication cost versus cache size when average waiting time is 40000 time slots.

V. Conclusion

We investigate the trade-off between the average waiting time and communication cost formally via a stochastic optimization problem introduced in Section III. We show that although deriving the exact solution to the optimization problem is hard, our proposed heuristic solutions (i.e., mapping algorithms) can manage this trade-off in different scenarios. While our first algorithm is suitable for demands with different delivery delay requirements, second and third schemes achieve better trade-off curves. Our results show that by sacrificing a small amount of communication cost one can arrive at a balanced network load.

Algorithm	Average Number of Queries Required
Algorithm 1 (PSS)	$\zeta \frac{LM}{N}$
Algorithm 2 (WMC)	$\frac{LM}{N}$
Algorithm 3 (MCS)	Δ

TABLE I
Average query cost of each proposed mapping algorithm.

References

[1] Cisco, "Cisco visual networking index: Forecast and methodology, 2014-2019 white paper," online: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html, Accessed: 2016-10-06.

[2] E. Nygren, R. K. Sitaraman, and J. Sun, "The akamai network: A platform for high-performance internet applications," *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 3, pp. 2–19, Aug. 2010.

[3] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Comput. Netw.*, vol. 57, no. 16, pp. 3128–3141, Nov. 2013.

[4] F. L. Presti, N. Bartolini, and C. Petrioli, "Dynamic replica placement and user request redirection in content delivery networks," in *IEEE International Conference on Communications (ICC)*, vol. 3, May 2005, pp. 1495–1501.

[5] C. Chen, Y. Ling, M. Pang, W. Chen, S. Cai, Y. Suwa, and O. Altintas, "Scalable request routing with next-neighbor load sharing in multi-server environments," in *19th International Conference on Advanced Information Networking and Applications (AINA 2005), 28-30 March 2005, Taipei, Taiwan*, 2005, pp. 441–446.

[6] S. Manfredi, F. Oliviero, and S. P. Romano, "A distributed control law for load balancing in content delivery networks," *IEEE/ACM Trans. Netw.*, vol. 21, no. 1, pp. 55–68, 2013.

[7] F. Chen, R. K. Sitaraman, and M. Torres, "End-user mapping: Next generation request routing for content delivery," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, ser. SIGCOMM '15, 2015, pp. 167–181.

[8] L. Wang, V. Pai, and L. Peterson, "The effectiveness of request redirection on cdn robustness," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 345–360, Dec. 2002.

[9] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations," *SIAM J. Comput.*, vol. 29, no. 1, pp. 180–200, 1999.

[10] M. Adler, S. Chakrabarti, M. Mitzenmacher, and L. E. Rasmussen, "Parallel randomized load balancing," *Random Struct. Algorithms*, vol. 13, no. 2, pp. 159–188, 1998.

[11] F. Xia, A. M. Ahmed, L. T. Yang, and Z. Luo, "Community-based event dissemination with optimal load balancing," *IEEE Trans. Computers*, vol. 64, no. 7, pp. 1857–1869, 2015.

[12] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, no. 10, pp. 1094–1104, 2001.

[13] M. Mitzenmacher, A. W. Richa, and R. Sitaraman, "The power of two random choices: A survey of technique and results," *In Handbook of Randomized Computation Volume 1*, pp. 255–312, 2001.

[14] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: evidence and implications," in *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, Mar 1999, pp. 126–134 vol.1.

[15] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system," in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '07. New York, NY, USA: ACM, 2007, pp. 1–14.

[16] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Information Theory*, vol. 62, no. 2, pp. 849–869, 2016.

[17] W. Winston, "Optimality of the shortest line discipline," *Journal of Applied Probability*, vol. 14, no. 1, pp. 181–189, 1977.