Joint Optimization of Caching and Transport in Proactive Edge Cloud

Stefania Sardellitti, Francesca Costanzo, and Mattia Merluzzi,
Sapienza Univ. of Rome, DIET Dept, Via Eudossiana 18, 00184, Rome, Italy
e-mail: {stefania.sardellitti, francesca.costanzo, mattia.merluzzi}@uniroma1.it

***Abstract*—Our goal in this paper is to devise a strategy for finding the optimal trade-off between the transport and caching energy costs associated to the delivery of contents in information networks. The proposed strategy is proactive with respect to the users' requests, as contents are pre-fetched depending on the distribution of their (estimated) popularity. In particular, we propose a k-center dominating set strategy to find the optimal clustering and then locate the best places to store/replicate the most popular contents. Then we develop a dynamic energy-efficient, strategy that jointly optimizes caching and delivery costs within each cluster. Although the formulated problem is a binary problem, we will show as it can be solved for moderate size networks by using efficient solvers. The performance gain reached through the proposed proactive strategy are then assessed by numerical results.**

***Index Terms*—Proactive content delivery, in-network caching, energy efficiency, information centric networking.**

I. Introduction

In-network content caching has received considerable attention in recent years as an effective way to address the explosive growth in Internet traffic of content access to sources such as YouTube, Netflix, Bit Torrent, etc. This rapid increase of content delivery in Internet has motivated the development of novel networking paradigms such as Information Centric Networking (ICN) that integrates content delivery as a native network feature and then it is better suited for efficiently accessing and distributing contents [1], [2]. One of the main benefits of ICN is to reduce user content access delay and network bandwidth usage by storing the contents at the network edge close to the end user. ICN is based on named data objects, for example, web pages, videos, documents, or other pieces of information. In contrast, current networks are host-centric since communication is based on named hosts, for example, web servers, PCs, mobile handsets. In ICN network, devices are equipped with storage capabilities to cache content as it is request by end user, so that access ICNs are able to cache the most popular content locally and store them near the users in the network. Therefore every node actively contributes in content caching to reduce the network congestion, the access delay and origin servers' load [3], [4]. Several content caching strategies have been proposed to maximize local hit rate, or the fraction of requests served by a given cache. The optimal placement of information objects in content delivery networks minimizing the access cost is investigated in [5]. A game theoretic approach to jointly find the caching and pricing strategy for popular content delivery was proposed in [6]. Some works [7], [8] address the problem of finding, in a static way, jointly the optimal placement and routing of information objects in cache networks to minimize the network energy consumption. In [7] the authors investigated the minimum energy consumption problem in content-centric networking (CCN) for optimal cache locations. The problem of finding the optimal content access delay minimizing the request routing and content caching in heterogeneous networks has been investigated in [9]. Clearly, an effective caching strategy builds significantly on the ability to learn and predict users' behaviors. This capability lies at the foundation of *proactive caching* [10] and it motivates the need to merge future networks with big data analytics [11]. An alternative approach to proactive caching, based on reinforcement learning to learn file popularity across time and space, was recently proposed in [12]. The authors in [13] developed a strategy to minimize the long-term average energy cost of content delivery based on Markov decision process. Different distributed dynamic content replacement strategies that refresh the caches contents as they travel through the network have been proposed in [14], [15]. The work in [15] considered the problem of finding the time evolution of the optimal placement and routing of contents which minimizes the sum of the transport and caching energies.

In this work we consider a joint proactive caching and routing strategy aiming at minimizing the sum of the caching and transport energy consumption in edge cloud networks [16] where the objects to be delivered are stored at edge nodes. By hinging on the strategy proposed in [15], we devise a proactive in-network caching method based on an-online popularity learning of the object contents. In particular, the cost of caching the information objects comes to depend dynamically on the local and global popularity of the objects, by forcing nodes to host the most frequently requested contents. Furthermore, the decision on which locations have to be repository nodes which host permanently the object contents, depends on the object popularity and on the network topology. The nodes that are able to reach the others with the minimum paths have to be preferred in order to save as much as possible the transport energy consumption. Hence, in this paper we proposed a k-center dominating set strategy to find the optimal clustering and, then, properly select the most convenient nodes where replicate contents. Then, we jointly

This work has been supported by H2020 EUJ Project 5G-MiEdge, Nr. 723171.

ISBN 978-90-827970-1-5 © EURASIP 2018

optimize the transport and caching energy by formulating a binary optimization problem whose optimal solution is found by using an optimization solver based on the branch and bound algorithm.

II. Network model

Let us consider a transport network represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{K})$, composed of a set of nodes $\mathcal{V}$, a set of links $\mathcal{E}$, and a set of information objects $\mathcal{K}$, as illustrates in Fig. 1. We assume that contents can be permanently or temporarily stored over the nodes of this graph or travel through its edges. More specifically, in Fig. 1 we denote with disks the repository nodes where the contents are permanently stored, and with circles the nodes where contents may appear and disappear, according to users' requests and network resource allocation. To simplify our formulation we assume that all contents can be split into objects of equal size, identified by the index $k \in \mathcal{K}$. Each node and edge is characterized, respectively, by a storage and transport capability. We assume that time is slotted with slots of fixed duration $\Delta\tau$ and, we observe time frames, each composed of T time slots. At time slot n each node $u \in \mathcal{V}$ in the network can act as a repository node of a set of information objects $K_u[n] \subseteq \mathcal{K}$, and can request a set of information objects $Q_u[n] \subseteq \mathcal{K}$. Let us denote with $\mathbf{q}[n] \in \{0,1\}^{|\mathcal{V}||\mathcal{K}|}$ the request arrival process such that $q_u[k,n] = 1$ if node u requests object k at time n, and $q_u[k,n] = 0$ otherwise. The random process $\mathbf{q}[n]$ depends on the time evolution of the contents' popularity that is modelled as a Poisson process. The average arrival rate $P_{uk}[n]$ at node u for object k follows the Zipf distribution, i.e. [17]

$$P_{uk}[n] = \beta_u[n] \frac{r_{uk}[n]^{-\alpha_u(n)}}{\sum_{k=1}^{|\mathcal{K}|} r_{uk}[n]^{-\alpha_u(n)}} \tag{1}$$

where $\alpha_u(n)$ is the Zipf parameter, $\beta_u[n]$ represents the request rate of node u at time n, and $r_{uk}[n]$ is the rank of object k at node u at time n. From (1), we can associate to each time frame s, a local and global popularity measure, respectively, $\bar{P}^l_{uk}[s]$ and $\bar{P}^g_k[s]$, defined as

$$\bar{P}^l_{uk}[s] = \frac{\sum_{n=1}^{T} q_u[k,n]}{\sum_{n=1}^{T}\sum_{k=1}^{|\mathcal{K}|} q_u[k,n]} \tag{2}$$

and

$$\bar{P}^g_k[s] = \frac{\sum_{n=1}^{T}\sum_{u=1}^{|\mathcal{V}|} q_u[k,n]}{\sum_{u=1}^{|\mathcal{V}|}\sum_{n=1}^{T}\sum_{k=1}^{|\mathcal{K}|} q_u[k,n]}. \tag{3}$$

Therefore, we assume that the objects popularity dynamically evolves in time according to the following local and global rules

$$P^l_{uk}[s] = \bar{P}^l_{uk}[s] + \sum_{m=-\infty}^{s-1} \eta^{s-1-m} P^l_{uk}[m] \tag{4}$$

$$P^g_{k}[s] = \bar{P}^g_{k}[s] + \sum_{m=-\infty}^{s-1} \eta^{s-1-m} P^g_{k}[m] \tag{5}$$

where $\eta \in (0,1)$ is a forgetting factor, taking into account all the previous probability, and whose value can be assigned in order to weight differently the popularity evolution. The object popularity plays a fundamental role in proactive caching since, as we will see in the sequel, it enables us to associate a cost to the caching depending on the object requests time evolution.

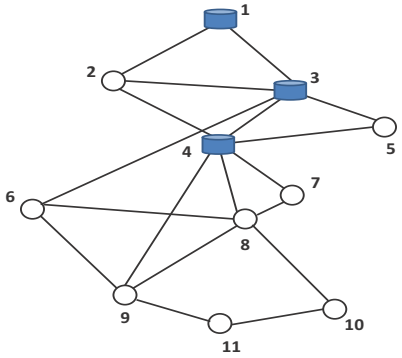

Fig. 1: New Jersey LATA optical network.

Hence, given a network, we can define a vertex signal over its nodes and an edge signal over its edges. The vertex signal $s_u[k,n]$ is a binary signal defined $\forall u \in \mathcal{V}$ as:

$$s_u[k,n] = \begin{cases} 1, \text{ if content } k, \text{ at time } n, \text{ is stored on node } u \\ 0, \text{ otherwise.} \end{cases}$$

The overall information objects stored on node u, at time n, can be written as $S_u[n] := \sum_{k=1}^{|\mathcal{K}|} s_u[k,n]$. Similarly, we can observe on the edges of the graph a transport flow signal, defined as a binary signal, i.e., $\forall uv \in \mathcal{E}$ we have

$$t_{uv}[k,n] = \begin{cases} 1, & \text{if content } k, \text{ at time } n, \text{ is transported} \\ & \text{over link } uv \text{ from node } u \text{ to node } v \\ 0, & \text{otherwise.} \end{cases}$$

Since each link uv at time n can transport several object information, the amount of transported content becomes $T_{uv}[n] := \sum_k t_{uv}[k,n]$. Typically, each content may be hosted on every node and moved whenever convenient to another location, by meeting the storage and capacity constraints on the nodes and edges of the network. Therefore, both $S_u[n]$ and $T_{uv}[n]$ belong to the following box constraints

$$0 \le S_u[n] \le S_u, \;\; 0 \le T_{uv}[n] \le T_{uv}, \tag{6}$$

where S_u is the storage capability of node u, whereas T_{uv} is the transport capacity of link uv. One of the main characteristic of caching is that it is fundamentally a dynamic process so that cached contents can be placed in some nodes and removed from others. We assume in our network model that there are only some repository nodes, for instance nodes 1,3,4 in Fig. 1, that permanently keep contents or may access to a content delivery network. Moreover, each content is hosted in at least one repository node. Although an object content $k \in \mathcal{K}$ may be stored at any time slot n in several repository locations, the

ISBN 978-90-827970-1-5 © EURASIP 2018

cost in keeping contents in a node can be high, especially in the case where the content is not frequently requested. Our goal in this work is to devise a proactive caching strategy which minimizes jointly the caching and transportation cost by taking into account that the cost of caching strictly depends on the popularity of the requested contents. Additionally, we propose a strategy to select the repository nodes depending on both the content popularity and the nodes centrality. Then, nodes having the minimum distance in terms of hops from all others, has to be chosen as repository nodes in order to save as much as possible the transport cost.

III. Tradeoff between storage and transport

One of the central challenges associated to caching is to strike the best tradeoff between the cost of storing a content in the network, possibly proactively, and the cost of transporting (delivering) the content when and where requested. The first question to answer is about the optimal number of replicas of a given content. Intuitively speaking, the more copies of a file we have, the smaller will be the delivery time, and viceversa. To make this intuition formal we indicate the optimal number of replicas as a function of the storage/transportation costs. Given a network with N nodes, whose topology is represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we denote by $\boldsymbol{B}$ the $N \times N$ matrix whose entry B_{ij} contains the length of the shortest path between nodes i and j.

If $P_j(\ell)$ is the probability that content ℓ will be requested in the area served by access node j, the average (expected) energy cost needed to deliver content ℓ from node i to the rest of the network is

$$T_i(\ell) = \sum_{j=1}^{N} B_{ij} P_j(\ell) T \tag{7}$$

where T is the unit cost to transport a content object over one link. If we wish to minimize this average cost and we can store content ℓ in only one location, clearly the best location is the one that minimizes the above cost, i.e.

$$i^*(\ell) := \underset{i \in \{1,\dots,N\}}{\operatorname{argmin}} \; T_i(\ell). \tag{8}$$

Correspondingly, in case of storing content ℓ in a single cache, the overall cost for delivering this content is

$$E_{i^*}(\ell) = S + T_{i^*}(\ell), \tag{9}$$

where i^* is computed using (8) and S is the energy cost for storing a content unit. Let us consider now the more interesting case where we wish to cache the same content in multiple places and we ask ourselves about how many times to replicate the same content and where to put these replicas. A possible strategy is the following. If we want to replicate M times, we can split the overall network in M clusters, individuate a cluster head and place the content in the cluster head. Different clustering techniques can be used, depending on the chosen optimization criterion [18]: 1) given a number M of clusters, find the cluster heads and the clusters that minimize the distance of each node from its cluster head; 2) minimize the number of clusters guaranteeing a maximum distance k between a node in the cluster and the corresponding cluster head. An important aspect, useful for the identification of the best way to build clusters, is the concept of *dominating set*. A dominating set of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a subset $\mathcal{D} \subseteq \mathcal{V}$, such that every node $v \in \mathcal{V}$ is either in $\mathcal{D}$ or adjacent to a vertex of $\mathcal{D}$. More specifically, a subset $\mathcal{D} \subseteq \mathcal{V}$ such that every node $v \in \mathcal{V}$ is within distance k neighborhood of some vertex of $\mathcal{D}$ is called a k-distance dominating set. If we want to ensure that a content will be delivered within at most k hops, the optimal clustering strategy consists then in finding the k-distance dominating sets and replicate the content in the nodes belonging to the dominating set. Alternatively, if we fix the number of clusters, we can look for the clustering method that minimizes the maximum distance of each node from its cluster head. This is an NP-hard problem, but there exist heuristics to find approximate solutions in polynomial time. In this paper, we used the approach proposed in [19]. If we denote by $\mathcal{C}_m$ the m-th cluster, with $m = 1, \dots, M$, the overall cost for storing and delivering content ℓ using M caches is then

$$E(\ell) = MS + T \sum_{m=1}^{M} \sum_{j \in \mathcal{C}_m} B_{mj} P_j(\ell), \tag{10}$$

letting j to vary within each cluster. We may expect that, as M increases, the storage cost increases linearly, whereas the transportation cost decreases. Then, we may expect to find a unique optimal value of M that minimizes the overall delivery cost. This optimal value depends on the ratio between the cost of storing and the cost of transporting a content. A numerical example is reported in Fig. 2 where we plot the total energy consumption for storing the most popular content, among 200, in a network of 300 nodes, versus the number of possible clusters M, for $S = 0.2\,T$. We can observe that for $M = 60$ there exists an optimal trade-off between the cost of duplicate the content in more cluster heads and the cost of deliver it.

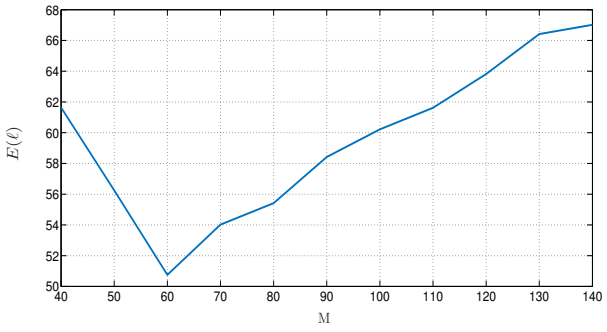

Fig. 2: Total energy cost versus the number of clusters M.

IV. Proactive joint caching and transport optimization

The fundamental issue in caching problem is to dynamically allocate all contents by deciding, according to the users' requests, when and where place all contents, how transport

ISBN 978-90-827970-1-5 © EURASIP 2018

them in the network and when a node has to release them to save memory. Then, the decision for caching content k at node u, at time slot n, has to depend on both the energy spent for storing the object, and the energy spent to transport it from its current position to the user who requested it. By making the caching cost dependent on the time evolution of the popularity of the requested contents, the caching strategy becomes *proactive* and *context-aware*. We define the energy cost for storing a content k on node u during T consecutive time slots, in the time window $[n'-T+1, n']$, as

$$E_{st}[n'] = \sum_{n=n'-T+1}^{n'} \sum_{k\in\mathcal{K}} \sum_{u\in\mathcal{V}} s_u[k,n] c_u[k,n], \qquad (11)$$

where $c_u[k,n]$ is the time-varying energy cost for keeping content k on node u at time n. For instance, from equations (4), (5), we can define the caching cost of content k at node u at time n as

$$c_u[k,n] = \frac{\gamma}{1+P^l_{uk}[n]} + \frac{1-\gamma}{1+P^g_k[n]} \qquad (12)$$

where $\gamma \in [0,1]$ is the trade-off coefficient between the local and global popularity costs. Note that the cost $c_u[k,n]$ is low for contents with the highest popularity to encourage the storage of these frequently requested contents.
Then, we can define the cost associated to the content transport as

$$E_{tr}[n'] = \sum_{n=n'-T+1}^{n'} \sum_{k\in\mathcal{K}} \sum_{uv\in\mathcal{E}} t_{uv}[k,n] c_{uv}[k], \qquad (13)$$

where $c_{uv}[k]$ represents the energy cost to transfer content k over link uv. When content k is requested by user u, we can associate to the user request a maximum delivery time $D_u[k]$ within which the request must be solved.

We denote by $\mathcal{N}_u$ the set of nodes that are one hop away from node u, i.e. the set of its neighbors. As discussed above, a fundamental issue in proactive caching is to select the repository nodes where the contents have to be stored as long as the popularity does not change. To make this choice *proactive* we associate at each node u a probabilistic measure of its centrality

$$w_u[k] = \frac{\sum_{uv\in\mathcal{E}} B_{uv} P_v[k]}{\sum_{v\in\mathcal{V}} P_v[k]} \qquad (14)$$

where B_{uv} is the length of the shortest path between nodes u and v and $P_v[k]$ is the average popularity of object k at node v. Therefore, according to (14) we can permanently store each object k in the node u where $w_u[k]$ takes its minimum value, i.e. in the node having the average minimum number of hops from the nodes requiring content k. Hence, we proactively define the set of repository nodes $\mathcal{V}_p$ where each $u \in \mathcal{V}_p$ hosts a set of information objects $\mathcal{K}^p_u$. The state of the network, at time slot n, is represented by the vector $\mathbf{x}[n] := [\mathbf{s}[n]; \mathbf{t}[n]]$, with $\mathbf{s}[n] := (s_u[k,n])_{\forall u,k}$ and $\mathbf{t}[n] := (t_{uv}[k,n])_{\forall k, uv\in\mathcal{E}}$. Then, we define $\mathbf{x}_T[n'] := [\mathbf{x}[n'-T+1]; \ldots; \mathbf{x}[n']]$ as the state vector during T consecutive time slots. Hence, our objective becomes to minimize with respect to the state vectors the sum of the caching and transport energies given in (11) and (13), i.e.

$$\sum_{n'=0}^{N_f} \lambda E_{tr}(\mathbf{x}_T[n']) + E_{st}(\mathbf{x}_T[n']) \qquad (15)$$

where N_f is the number of observed time frames and λ is a positive parameter controlling the ratio between transport and storage energy costs. By extending to the proactive caching the approach proposed in [15], we simplify our problem by decoupling the objective function in (15) over each frame n'. Then our goal is to find the state vector $\mathbf{x}_T[n']$, for each frame n', which minimizes the cost function

$$E_T(\mathbf{x}_T[n']) := \lambda E_{tr}(\mathbf{x}_T[n']) + E_{st}(\mathbf{x}_T[n']) \qquad (16)$$

where we assumed in (13) a unit transport cost $c_{uv}[k]$. Then, for each time frame, we can formulate the proactive caching optimization problem as

$$\begin{aligned} \hat{\mathbf{x}}_T = \arg\min_{\mathbf{x}_T} \quad & \lambda E_{tr}(\mathbf{x}_T) + E_{st}(\mathbf{x}_T) \\ \text{s.t.} \quad & \mathbf{x}_T \in \mathcal{X} \end{aligned} \qquad (\mathcal{P})$$

where the constraint set $\mathcal{X}$ is defined as

$$\mathcal{X} \doteq \begin{cases} (a)\ \ q_u[k,n] \le s_u[k,n] + \sum_{v\in\mathcal{N}_u} \sum_{j=0}^{D_u[k]} t_{vu}[k,n+j] \\ (b)\ \ s_u[k,n] \le s_u[k,n-1] + \sum_{v\in\mathcal{N}_u} t_{vu}[k,n-1] \\ (c)\ \ t_{vu}[k,n] \le s_v[k,n-1] + \sum_{w\in\mathcal{N}_v} t_{wv}[k,n-1] \\ (d)\ \ s_u[k,n] = 1, \forall k \in K^p_u,\ s_u[k,0] = 0, k \notin K^p_u \\ (e)\ \ S_u[n] \le S_u \\ (f)\ \ T_{vu}[n] \le T_{vu} \\ (g)\ \ s_u[k,n] \in \{0,1\},\ t_{uv}[k,n] \in \{0,1\}, \end{cases}$$

$\forall u \in \mathcal{V}, vu \in \mathcal{E}, k \in \mathcal{K}, n \in [n'-T+1, n']$.

The above constraints reflect the storage and transport conservation flow constraints [15]. In more detail:(a) assures that if object k is requested by node u at time slot n, then k either is already present in the cache of node u at time n or it has to be transported to node u from a neighbor node $v \in \mathcal{N}_u$ within $D_u[k]$ time slots; (b) ensures that if k is being cached at node u at time n, then k either was in the cache of u at time $n-1$ or was received by node u from a neighbor node $v \in \mathcal{N}_u$ at time $n-1$; (c) imposes that if object k is delivered to node u from a neighbor node $v \in \mathcal{N}_u$ at time n, then this object either was in the cache of v at time $n-1$ or was transferred to u from a neighbor node $w \in \mathcal{N}_v$ at time $n-1$; (d) forces the initial condition constraints that assure a proactive selection of the repository nodes that always store the objects in $\mathcal{K}^p_u$, and at $n=0$ nothing else; (e) and (f) force the storage and transport capacity constraints; (g) states the binary nature of the storage and transport variables. Note

ISBN 978-90-827970-1-5 © EURASIP 2018

that problem $\mathcal{P}$ is hard to be solved for large network size because of its combinatorial complexity. However, how we will show next, it can be solved efficiently, for moderate size networks, by using a numerical solver based on the branch and bound algorithm. To numerically test the efficiency of the proposed caching strategy we consider the 11 nodes New Jersey LATA network illustrated in Fig. 1. We use to solve problem $\mathcal{P}$ the cvx binary solver Mosek based on the branch and bound algorithm. We consider the delivery of $|\mathcal{K}| = 10$ contents during time frames of $T = 15$ slots, each of duration $\Delta\tau = 1$. The maximum delivery $D_u[k]$, is set equal to 4, that is the maximum distance in number of hops between two nodes in the network. The objects popularity follows a Zipf distribution with a high parameter value $\alpha_u = 2.5$, $\forall u$, to increase the skewness of the popularity profile. The forgetting factor is set to $\eta = 0.7$ and the trade-off coefficient for the popularity to $\gamma = 0.8$. We averaged the final results over $N_f = 50$ time frames. In Fig. 3 we report the transport gain in terms of number of hops reduction with respect to the non-proactive caching strategy [15] where the content popularity does not affect the optimization process. The transport gain

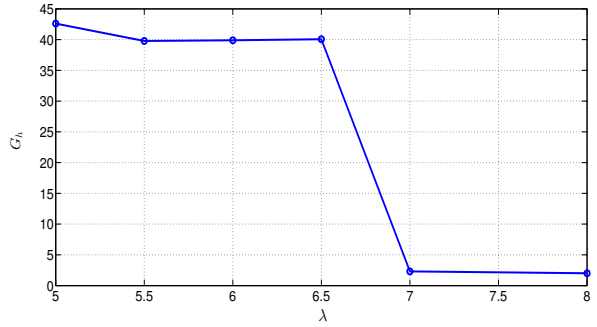

Fig. 3: Number of hops gain G_h versus λ.

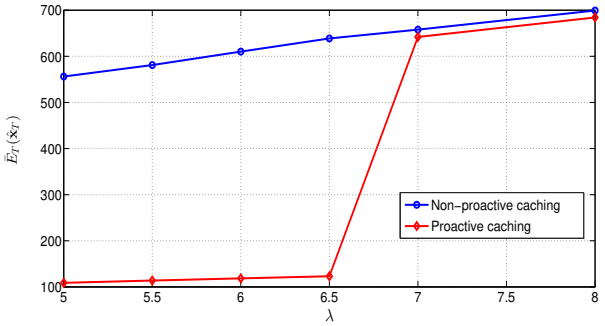

Fig. 4: Total average energy cost comparison versus λ.

is defined as $G_h \doteq n_h^n - n_h^p$ where n_h^n and n_h^p are the average number of hops, respectively, in the non-proactive and in the proactive algorithms. From Fig. 3 it can be observed as G_h increases when the transport cost λ decreases, since, in this case, the objects transport is favoured by the proactive caching. Finally, in Fig. 4 we plot the energy consumption comparison between the proactive and non proactive algorithms versus the parameter λ. Note that proactivity leads considerable energy savings for low λ values, taking benefit from the optimal transport strategy.

In summary, in this paper we showed as a cluster-based strategy, aided by the content popularity, can be used to properly select the nodes where replicate object contents. A joint proactive caching and transport optimization strategy has been proposed to favour caching of the most popular contents and to decrease the transport cost, by taking advantage of the reduced number of hops needed to delivery contents.

REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. 5th Int. Conf. Emerg. Netw. Exper. Technol.* ACM, 2009, pp. 1–12.

[2] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 26–36, July 2012.

[3] M. Zhang, H. Luo, and H. Zhang, "A survey of caching mechanisms in information-centric networking," *IEEE Commun. Surveys Tut.*, vol. 17, no. 3, pp. 1473–1499, 2015.

[4] W. Wang and et al., "CRCache: Exploiting the correlation between content popularity and network topology information for ICN caching," in *Proc. IEEE Int. Conf. Commun. (ICC 2014)*, 2014, pp. 3191–3196.

[5] I. D. Baev and R. Rajaraman, "Approximation algorithms for data placement in arbitrary networks," in *Proc. 20th Ann. ACM-SIAM Symp. Disc. Alg.*, 2001, pp. 661–670.

[6] M. Hajimirsadeghi, N. B. Mandayam, and A. Reznik, "Joint caching and pricing strategies for popular content in information centric networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 3, pp. 654–667, Mar. 2017.

[7] N. Choi, K. Guan, D. C. Kilper, and G. Atkinson, "In-network caching effect on optimal energy consumption in content-centric networking," in *Proc. IEEE Int. Conf. Commun. (ICC 2012)*, June 2012, pp. 2889–2894.

[8] A. Khreishah, H. B. Salameh, I. Khalil, and A. Gharaibeh, "Renewable energy-aware joint caching and routing for green communication networks," *IEEE Syst. J.*, vol. 12, no. 1, pp. 768–777, Mar. 2018.

[9] M. Dehghan, A. Seetharam, B. Jiang, T. He, T. Salonidis, J. Kurose, D. Towsley, and R. Sitaraman, "On the complexity of optimal routing and content caching in heterogeneous networks," in *Proc. IEEE Conf. Compu. Commun. (INFOCOM 2015)*, Apr. 2015, pp. 936–944.

[10] E. Baştuğ, M. Bennis, E. Zeydan, M. A. Kader, I. A. Karatepe, A. S. E., and M. Debbah, "Big data meets telcos: A proactive caching perspective," *J. Commun. Netw.*, vol. 17, no. 6, pp. 549–557, Dec. 2015.

[11] E. Zeydan, E. Baştuğ, M. Bennis, M. A. Kader, I. A. Karatepe, A. S. Er, and M. Debbah, "Big data caching for networking: Moving from cloud to edge," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 36–42, Sept. 2016.

[12] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, "Optimal and scalable caching for 5G using reinforcement learning of space-time popularities," *IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 1, pp. 180–190, Feb. 2018.

[13] S. O. Somuyiwa, A. György, and D. Gündüz, "Energy-efficient wireless content delivery with proactive caching," in *2017 15th Int. Symp. Model. Optim. Mob., Ad Hoc, Wir. Netw. (WiOpt)*, May 2017, pp. 1–6.

[14] J. Wang, "A survey of web caching schemes for the Internet," *ACM SIGCOMM Compu. Commun. Rev.*, vol. 29, no. 5, pp. 36–46, 1999.

[15] J. Llorca, A. M. Tulino, M. Varvello, J. Esteban, and D. Perino, "Energy efficient dynamic content distribution," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2826–2836, Oct. 2015.

[16] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[17] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. IEEE 18th Annu. Joint Conf. Comput. Commun. Soc. (INFOCOM'99)*, vol. 1, 1999, pp. 126–134.

[18] Y. Chen, A. Liestman, and J. Liu, "Clustering algorithms for ad hoc wireless networks," *Ad Hoc Sensor Netw.*, vol. 28, p. 76, 2004.

[19] B. Robič and J. Mihelič, "Solving the k-center problem efficiently with a dominating set algorithm," *J. Comput. Inform. Technol.*, vol. 13, no. 3, pp. 225–234, 2005.

ISBN 978-90-827970-1-5 © EURASIP 2018