Proactive Caching in Mobile Networks with Delay Guarantees

Yantong Wang, Gao Zheng, Vasilis Friderikos
Center for Telecommunications Research, Department of Informatics
King's College London, London WC2B 4BG, U.K.
E-mail:{yantong.wang, gao.zheng, vasilis.friderikos}@kcl.ac.uk

***Abstract*— The explosive growth of mobile data traffic and the envisioned delay sensitive applications in 5G networks ranging from high definition video streaming with strict playout deadlines to multi modal tactile/haptic with kinaesthetic feedback that require some form of edge cloud cache support makes mobility a challenge. In this paper, we propose a Proactive Caching with Delay Guarantees (PCDG) approach to enhance the supporting of seamless mobility within 5G networks that are Information-Centric Networking (ICN)-aware. The proposed scheme is designed to cache contents into a set of potential edge clouds with delay guarantees and to achieve a trade-off among caching, redirection and missing cost. In particular, this approach consider the delay constraints in mobile network, especially the queuing delay in network links and edge clouds, which are modeled as M/M/1 and M/M/c queuing systems respectively. We formulate and linearize this problem as a Mixed Integer Linear Programming (MILP) model and compare the performance with other techniques. The result obtained from simulation reveal that the proposed PCDG scheme lead to a significantly lower total cost and higher satisfied probability albeit higher computational/complexity cost.**

***Index Terms*— Proactive Caching, Information-Centric Network, Mixed Integer Linear Programming, Queuing Theory**

I. Introduction

IT has been forecast that by 2021, data traffic from mobile devices will constitute more than 63% of the total aggregate traffic on the Internet [1]. This envisioned explosive growth of mobile traffic makes mobility support a challenge in traditional network architecture, since mobility management at the network layer poses a number of challenges such as tunnelling, encapsulation and non-optimal path redirection [2]. As an evolutionary framework, Information-Centric Networking (ICN) has been proposed where the use of information-based naming instead of host-based IP address, ease the support of seamless mobility [2], [3]. Within this emerging framework, a number of approaches have been suggested to enhance mobility support namely the so-called subscriber (e.g. [4]) and publisher (e.g. [5]) frameworks. Hereafter, the subscriber mobility model is a salient assumption of the proposed pro-active caching model.

When a mobile user[1] (subscriber) changes its point of attachment the supported caching methods can be taxonomized as follows [4] and [6]: reactive approaches [7], durable subscriptions [8] and proactive approaches [9]. Compared with the so-called reactive approaches and durable subscriptions the proactive approaches make a trade-off between available storage capacity for caching and latency. As alluded in [6] the holy grail in proactive approaches relate to the question of *what to cache* and *where to cache*. Some relevant previous works [10] and [11] use different learning methods to tackle these issues, such as transfer learning and online learning; the aim is to estimate content popularity and place in an optimal manner popular content into cache hosts to increase cache hit ratio. To select the set of routers to cache popular content the work in [4] present a Selective Neighbor Caching (SNC) approach that finds efficient one-hop neighbors in regarding to the current edge-cloud to cache content. As an extension to the SNC scheme, [12] redefines the set of neighbor edge-clouds by taking the mobility of the user into consideration. The work in [6] establish a formal optimization framework as with respect to pro active caching using a probabilistic model regarding future points of attachments of the users. These works try to keep a balance between cache cost and delay time with cache capacity constraints, but they do not take explicitly latency constraints into consideration that are caused by queuing in the network and/or the edge-clouds. To illustrate the case, let's consider the following extreme scenario: there is a neighbor node (edge cloud) which has infinite caching space but limited service ability, this node would be selected by the majority of mobile users as their cache host by the techniques in [4], [6], [12]. However, this will cause an increased latency since these previous works do not consider the node ability in terms of queueing to respond to these requests.

To explicitly consider latency constraints we propose a Proactive Caching with Delay Guarantees (PCDG) approach to enhance mobility support in an ICN-enabled mobile network. The objective is to minimize a cost function that captures caching, redirection and content miss cost. A key differentiation is that the proposed scheme considers caching in any potential edge cloud instead of being limited to neighboring nodes only. To this end, more network-wide resources, like caching storage space and available Virtual Machines (VMs) could be explored to support mobile users in the case of congestion episodes. If the mobile user hits the cache, he/she will experience negligible routing but the system needs to pay a cost for caching the content. While a subscriber moves to a destination that does not hold the cache, the requests of the subscriber are redirected to the closest proactive caching point such that a relative routing cost may be paid yet it is better than totally missing the cache. Either caching or redirecting a request, the delay tolerant requirement of the request must be satisfied, otherwise

[1]Hereafter we will be using the terms 'mobile user' and 'subscriber' interchangeably.

some penalty cost will be paid. In short, PCDG provides an optimal proactive cache allocation which jointly consider caching and routing under the condition if there exists a solution that can satisfy the request delay Quality of Service (QoS). In addition, PCDG provides detailed analysis of delay estimation, focusing on queuing delay which can vary based on the utilization levels, by amalgamating queuing theory results into integer linear programming models. More specifically, network links are modeled using the M/M/1 queuing model and service capacity in terms of available VMs of each potential candidate caching node are modeled using the M/M/c queuing model.

II. Modelling and Mathematical Programming

A mobile core/access network is modelled as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$, where $\mathcal{V}$ denotes the set of vertices and $\mathcal{L}$ is the set of links. We define a set $\mathcal{E} \subseteq \mathcal{V}$ that consist of the potential candidate nodes, i.e., edge clouds (ECs) where information can be hosted [2]. By $\mathcal{D} \subseteq \mathcal{V}$, we define the set of potential destinations that mobile users might move due to their mobility; this information is assumed to be accurately known using historical data that an operator can explore. We note that in the general case it is possible that the following holds $\mathcal{E} \cap \mathcal{D} \neq \emptyset$.

For network modelling reasons and without loss of generality, we assume that each mobile user is associated with a single request flow, and to this end, we define with $k \in \mathcal{K}$ the set of flows to traverse the mobile network. Each flow k has the following four associated properties: S_k, which is the size of cache items for flow k;R_k, the required transmission rate of flow k, δ_k the maximum delay tolerance for flow k and $P_{k,d}$ which encapsulate the probability for flow k to move to access router d, where $d \in \mathcal{D}$.

Similarly, for each potential edge cloud $e \in \mathcal{E}$:W_e is the total storage space in that node e, and with W_e^{re} we express the remaining cache space in edge cloud e. With c_e we denote the available number of virtual machines in edge cloud e and $\mu_{\epsilon,e}$ is the number of flow requests that each VM ϵ in edge cloud e could serve during the scope period of consideration. Each flow request could be serviced by a VM, and we assume without loss of generality that these are isolated, i.e., no degradation of the performance when multiple VMs run on the same bare metal hardware. In addition to that we assume that all VMs in a EC e have the same capabilities in terms of resources to respond to the various flow requests, i.e. $\mu_{\epsilon,e} = \mu_e, \forall \epsilon$. The key notations we used in this paper are summarized in Table I.

Based on the aforementioned network setting detailed in the previous section and in order to provide a mathematical programming framework we define the following binary decision variables,

$$x_{k,e} = \begin{cases} 1, & \text{if content for flow } k \text{ cached at EC } e \\ 0, & \text{otherwise} \end{cases}$$

[2] The term *potential candidate nodes* and *edge clouds* are used interchangeably in the rest of the paper

TABLE I
Summary of Main Notations Used In This Paper.

S_k	size of cache items for flow k
R_k	required transmission rate of flow k
δ_k	maximum delay tolerance for flow k
$P_{k,d}$	probability for flow k move to destination node d
W_e	total storage space in edge cloud e
W_e^{re}	remaining cache memory in edge cloud e
W_t^{re}	total remaining cache memory in network
c_e	available number of VMs in edge cloud e
μ_e	service rate of each VM in edge cloud e
$B_{l,d,e}$	binary matrix indicates whether link l is in the shortest path between destination node d and edge cloud e
C_k^{host}	cost of hosting the cache items for flow k
C_k^{cache}	cost of redirecting the cache node for flow k
C_k^{miss}	cost of missing cache for flow k
Q_k^{time}	penalty of not satisfying the time limitation
$x_{k,e}$	decision variable indicates whether content for flow k is placed at edge cloud e
$\pi_{k,d,e}$	decision variable indicates whether content for flow k retrieve from edge cloud e when get access to destination node d
$y_{k,l}$	decision variable indicates whether flow k passes link l

$$\pi_{k,d,e} = \begin{cases} 1, & \text{if } k \text{ required at } d \text{ and retrieve the cached} \\ & \text{content from EC } e \\ 0, & \text{otherwise} \end{cases}$$

We define the total cost with the following expression,

$$TC = \sum_{k \in \mathcal{K}} \left(\alpha \cdot C_k^{host} + \beta \cdot C_k^{cache} + \gamma \cdot C_k^{miss} \right) \quad (1)$$

where α, β and γ are the impact factors that control the weight of these three different price.

Specifically, C_k^{host} is the cost to host the content which is requested by flow k. This cost can be written as follows [4]:

$$C_k^{host} = \sum_{e \in \mathcal{E}} \frac{x_{k,e}}{1 - U_e} \quad (2)$$

and U_e is the cache utilization level at EC e. Note that in this paper, U_e is a variable depending on the content caching assignment (i.e., depend on the decision variable $x_{k,e}$),

$$U_e = \frac{W_e - W_e^{re} + \sum_{k \in \mathcal{K}} S_k \cdot x_{k,e}}{W_e} \quad (3)$$

Combining formula (2) and (3), C_k^{host} could be rewritten as:

$$C_k^{host} = \sum_{e \in \mathcal{E}} \frac{W_e}{W_e^{re} - \sum_{k \in \mathcal{K}} S_k \cdot x_{k,e}} \cdot x_{k,e} \quad (4)$$

C_k^{cache} express the redirected cost when the mobile user (i.e. flow k) connects to destination d but d does not cache the content:

$$C_k^{cache} = \sum_{d \in \mathcal{D}} \sum_{e \in \mathcal{E}} P_{k,d} \cdot C_{d,e}^{sp} \cdot \pi_{k,d,e} \quad (5)$$

where $C_{d,e}^{sp}$ is the cost of the shortest path between access point d and cache hosting EC e, which is calculated by the sum of link weights. Notably $C_{d,e}^{sp} = 0$ if $d = e$.

C_k^{miss} is the cost for flow k missing cache, i.e. when k move to access point d, but there is no such $\pi_{k,d,e}$ to retrieve

the caching content from e.

$$C_k^{miss} = \Big(1 - \sum_{d\in\mathcal{D}}\sum_{e\in\mathcal{E}} P_{k,d}\cdot\pi_{k,d,e}\Big)\cdot C_k^{pnt} \tag{6}$$

C_k^{pnt} is the penalty cost for k if its cache is missed.

The objective of this paper is to determine the optimal caching strategy that minimizes total cost TC. Expanding (1) based on (4) (5) and (6), then the total cost minimization problem can be formulated as:

$$\min_{x_{k,e},\pi_{k,d,e}} \quad TC \tag{7}$$

$$\text{s.t.} \quad \sum_{e\in\mathcal{E}} x_{k,e} \le 1, \quad \forall k\in\mathcal{K} \tag{7a}$$

$$\sum_{k\in\mathcal{K}} S_k\cdot x_{k,e} \le W_e^{re}, \quad \forall e\in\mathcal{E} \tag{7b}$$

$$\sum_{k\in\mathcal{K}}\sum_{e\in\mathcal{E}} S_k\cdot x_{k,e} \le W_t^{re} \tag{7c}$$

$$\pi_{k,d,e} \le x_{k,e}, \quad \forall k\in\mathcal{K}, d\in\mathcal{D}, e\in\mathcal{E} \tag{7d}$$

$$\pi_{k,d,e} \le M\cdot P_{k,d}, \quad \forall k\in\mathcal{K}, d\in\mathcal{D}, e\in\mathcal{E} \tag{7e}$$

$$\sum_{e\in\mathcal{E}} \pi_{k,d,e} \le 1, \quad \forall k\in\mathcal{K}, d\in\mathcal{D} \tag{7f}$$

$$t_k \le \delta_k, \quad \forall k\in\mathcal{K} \tag{7g}$$

$$x_{k,e}, \pi_{k,d,e} \in \{0,1\}, \quad \forall k\in\mathcal{K}, d\in\mathcal{D}, e\in\mathcal{E} \tag{7h}$$

where M is a sufficiently large number and t_k is the delay time from mobile user sending requirement to getting related contents in EC. Constraint (7a) limits the number of caching ECs for each flow. (7b) and (7c) show the cache capacity for individual and total EC respectively, where (7d), (7e) and (7f) enforce the redirected EC should host related contents, EC could not be retrieved if the probability of moving to relevant destination is 0 and the redirected path is unique. Moreover, (7g) impose the delay time for each mobile user should satisfy their delay tolerance.

In order to solve this optimization problem by existing mathematical tools, we need to transform the problem (7) into Mixed Integer Linear Programming (MILP) Model. It is worth noticing that the denominator of C_k^{host} contains decision variable $x_{k,e}$ in (4). Besides, the delay time t_k in constraint (7g) must be rewritten in an analytical form for the model to be possible to solve.

A. Delay Analysis

In this subsection, we determine the analytical form of t_k in the constraint (7g). According to [13], the delay which affect network performance can be divided into processing delay, transmission delay, propagation delay and queuing delay. Here different caching assignment would influence the queuing delay mainly, which could be divided into: (i) the link delay with tolerance δ^{link}, and it refers to the time required for data flows go across link; (ii) the EC delay with tolerance δ^{edge}, which means the time to access in the cache EC. Hereafter we focus on the queuing delay analysis and ignore the other kinds of delay. Then (7g) could be rewritten as:

$$t_k = t_k^{link} + t_k^{edge} \le \delta_k^{link} + \delta_k^{edge} \le \delta_k, \forall k\in\mathcal{K} \tag{8}$$

1) Link Delay: The link delay can be described as M/M/1 queuing model [14], where the flow arrives following a Poisson process and the serving time for this flow comes from anther Poisson process. According to Burke's theorem [15], the output of a M/M/1 queue is still follows Poisson distribution, so we could analyze each link independently:

$$t_k^{link} = \sum_{l\in\mathcal{L}} \frac{\sum_{e\in\mathcal{E}} B_{l,d,e}\cdot\pi_{k,d,e}\cdot(\sum_{k\in\mathcal{K}} R_k\cdot y_{k,l})}{C_l - \sum_{k\in\mathcal{K}} R_k\cdot y_{k,l}} \le \delta_k^{link}$$
$$\forall k\in\mathcal{K}, d\in\mathcal{D} \tag{9}$$

where $B_{l,d,e}$ shows the relationship between link and path, which could be generated from network topology by defining

$$B_{l,d,e} = \begin{cases} 1, & \text{if link } l \text{ in shortest path between } d \text{ and } e \\ 0, & \text{otherwise} \end{cases}$$

C_l is link l capacity and

$$y_{k,l} = \begin{cases} 1, & \text{if flow } k \text{ passes link } l \\ 0, & \text{otherwise} \end{cases}$$

which follows constraints below:

$$y_{k,l} \le \sum_{d\in\mathcal{D}}\sum_{e\in\mathcal{E}} B_{l,d,e}\cdot\pi_{k,d,e}, \quad \forall k\in\mathcal{K}, l\in\mathcal{L} \tag{10a}$$

$$M\cdot y_{k,l} \ge \sum_{d\in\mathcal{D}}\sum_{e\in\mathcal{E}} B_{l,d,e}\cdot\pi_{k,d,e}, \quad \forall k\in\mathcal{K}, l\in\mathcal{L} \tag{10b}$$

$$y_{k,l} \in \{0,1\}, \quad \forall k\in\mathcal{K}, l\in\mathcal{L} \tag{10c}$$

Constraints (10a) and (10b) enforce the link which flow passed should belong to a retrieved path, and vice versa.

2) Edge Cloud Delay: According to [16], the access to the VM at caching EC follows an M/M/c queue, where c is the number of VMs in EC, the processing time for each flow request at EC e is follow a Poisson distribution with average μ_e and the according arriving rate for flow follows another Poisson process with parameter λ_e. In other words, λ_e is the number of arriving flows in EC e per unit time i.e.

$$\lambda_e = \sum_{k\in K} x_{k,e}, \forall e\in\mathcal{E} \tag{11}$$

From queuing theory [14], the occupation rate ρ_e for EC e can be derived by

$$\rho_e = \frac{\lambda_e}{c_e\cdot\mu_e} \tag{12}$$

Then the waiting time in EC queue comes instantly from queuing theory.

$$t_k^{edge} = \frac{\rho_e(c_e\rho_e)^{c_e}}{\lambda_e c_e!(1-\rho_e)^2}\cdot p_e^0 \le \delta_k^{edge}, \forall k\in\mathcal{K}, e\in\mathcal{E} \tag{13}$$

and p_e^0 in (13) is the probability of 0 flows in EC e, which is represented by

$$p_e^0 = \Big[\sum_{k=0}^{c_e-1}\frac{(c_e\rho_e)^k}{k!} + \frac{(c_e\rho_e)^{c_e}}{c_e!(1-\rho_e)}\Big]^{-1}, \forall e\in\mathcal{E} \tag{14}$$

Finally, once the assignment of cache host $x_{k,e}$ is determined, the delay for accessing EC could be calculated by combining formula (11),(12),(13) and (14).

B. Linearizion of the Optimization Model

Though we get the analytical form of delay time t_k in constraint (7g), we still have the non-linear part in objective function (7). What's worse, by dividing the queuing delay into link delay and edge cloud delay, we introduce more non-linear formula in our model, such as (9) and (13). In this subsection, we use different liberalization tricks to transform the previous optimization problem into a MILP model.

1) Linearizion of Objective Function: For the purpose of linearizing the decision variable $x_{k,e}$ in denominator of C_k^{host} (4) in objective function (7), we define a new variable:

$$\chi_e = \frac{1}{W_e^{re} - \sum_{k\in\mathcal{K}} S_k \cdot x_{k,e}}, \forall e \in \mathcal{E} \tag{15}$$

This definition is equal to the constraints below:

$$W_e^{re} \cdot \chi_e - \sum_{k\in\mathcal{K}} S_k \cdot \chi_e \cdot x_{k,e} = 1, \quad \forall e \in \mathcal{E} \tag{16a}$$

$$\chi_e > 0, \quad \forall e \in \mathcal{E} \tag{16b}$$

so (4) becomes:

$$C_k^{host} = \sum_{e\in\mathcal{E}} \frac{W_e \cdot x_{k,e}}{W_e^{re} - \sum_{k\in\mathcal{K}} S_k x_{k,e}} = \sum_{e\in\mathcal{E}} W_e \cdot \chi_e \cdot x_{k,e} \tag{17}$$

Noticed that there is a product of two decision variables in (17), so we rewrite the model in terms of $\phi_{k,e}$ where:

$$\phi_{k,e} = \chi_e \cdot x_{k,e} = \begin{cases} \chi_e, & \text{if } x_{k,e} \text{ is 1} \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

and the constraints for $\phi_{k,e}$:

$$\phi_{k,e} \le \chi_e, \quad \forall k \in \mathcal{K}, e \in \mathcal{E} \tag{19a}$$

$$\phi_{k,e} \le M \cdot x_{k,e}, \quad \forall k \in \mathcal{K}, e \in \mathcal{E} \tag{19b}$$

$$\phi_{k,e} \ge M \cdot (x_{k,e} - 1) + \chi_e, \quad \forall k \in \mathcal{K}, e \in \mathcal{E} \tag{19c}$$

Now objective function (7) becomes:

$$TC^{new} = \sum_{k\in\mathcal{K}} \Big[\alpha \sum_{e\in\mathcal{E}} W_e \cdot \phi_{k,e} + \beta \sum_{d\in\mathcal{D}} \sum_{e\in\mathcal{E}} P_{k,d} \cdot C_{d,e}^{sp} \cdot \pi_{k,d,e} + \gamma C_k^{pnt} \cdot \big(1 - \sum_{d\in\mathcal{D}} \sum_{e\in\mathcal{E}} P_{k,d} \cdot \pi_{k,d,e}\big) \Big] \tag{20}$$

2) Linearisation of Link Queuing Delay: The link queuing delay formulation (9) is non-linear due to the denominator. In order to linearize it, we introduce a new decision variable z_l as the maximum delay tolerance for each link l, (9) can be transformed to:

$$t_k^{link} = \sum_{l\in\mathcal{L}} \sum_{e\in\mathcal{E}} B_{l,d,e} \pi_{k,d,e} \frac{\sum_{k\in\mathcal{K}} R_k y_{k,l}}{C_l - \sum_{k\in\mathcal{K}} R_k y_{k,l}} \le \sum_{l\in\mathcal{L}} \sum_{e\in\mathcal{E}} B_{l,d,e} (\pi_{k,d,e} \cdot z_l) \le \delta_k^{link}, \forall k \in \mathcal{K}, d \in \mathcal{D} \tag{21}$$

and the constraints for z_l:

$$\frac{\sum_{k\in\mathcal{K}} R_k y_{k,l}}{C_l - \sum_{k\in\mathcal{K}} R_k y_{k,l}} \le z_l, \quad \forall l \in \mathcal{L} \tag{22a}$$

$$z_l \ge 0, \quad \forall l \in \mathcal{L} \tag{22b}$$

By introducing a new constraint to keep the link queue stable:

$$C_l - \sum_{k\in\mathcal{K}} R_k y_{k,l} > 0, \forall l \in \mathcal{L} \tag{23}$$

Then we multiply $C_l - \sum_{k\in\mathcal{K}} R_k y_{k,l}$ on the each side of (22a), so it becomes:

$$\sum_{k\in\mathcal{K}} R_k y_{k,l} \le C_l z_l - \sum_{k\in\mathcal{K}} R_k (y_{k,l} z_l), \forall l \in \mathcal{L} \tag{24}$$

Notice that the products of two decision variables,i.e. $\pi_{k,d,e} z_l$ in (21) and $y_{k,l} z_l$ in (24), make these two formula non-linear. Similarly as the trick for linearising (17), we introduce two new decision variables $\psi_{k,l,d,e}$ and $\omega_{k,l}$ to replace the product $\pi_{k,d,e} z_l$ and $y_{k,l} z_l$ respectively, with constraints:

$$\psi_{k,l,d,e} \le z_l, \quad \forall k \in \mathcal{K}, l \in \mathcal{L}, d \in \mathcal{D}, e \in \mathcal{E} \tag{25a}$$

$$\psi_{k,l,d,e} \le M \cdot \pi_{k,d,e}, \quad \forall k \in \mathcal{K}, l \in \mathcal{L}, d \in \mathcal{D}, e \in \mathcal{E} \tag{25b}$$

$$\psi_{k,l,d,e} \ge M \cdot (\pi_{k,d,e} - 1) + z_l, \forall k \in \mathcal{K}, l \in \mathcal{L}, d \in \mathcal{D}, e \in \mathcal{E} \tag{25c}$$

$$\omega_{k,l} \le z_l, \quad \forall k \in \mathcal{K}, l \in \mathcal{L} \tag{25d}$$

$$\omega_{k,l} \le M \cdot y_{k,l}, \quad \forall k \in \mathcal{K}, l \in \mathcal{L} \tag{25e}$$

$$\omega_{k,l} \ge M \cdot (y_{k,l} - 1) + z_l, \quad \forall k \in \mathcal{K}, l \in \mathcal{L} \tag{25f}$$

$$\psi_{k,l,d,e}, \omega_{k,l} \ge 0, \quad \forall k \in \mathcal{K}, l \in \mathcal{L}, d \in \mathcal{D}, e \in \mathcal{E} \tag{25g}$$

At last, (21) and (24) could be rewritten as

$$\sum_{l\in\mathcal{L}} \sum_{e\in\mathcal{E}} B_{l,d,e} \psi_{k,l,d,e} \le \delta_k^{link}, \quad \forall k \in \mathcal{K}, d \in \mathcal{D} \tag{26}$$

$$\sum_{k\in\mathcal{K}} R_k y_{k,l} \le C_l z_l - \sum_{k\in\mathcal{K}} R_k \omega_{k,l}, \quad \forall l \in \mathcal{L} \tag{27}$$

3) Linearisation of Edge Cloud Queuing Delay: To linearize formulation(13), we combine (11)∼(14) and construct function $f(\lambda_e)$ as the difference between t_k^{edge} and δ_k^{edge}:

$$f(\lambda_e) = \frac{(\lambda_e)^{c_e}}{c_e!(\mu_e)^{c_e}} \Big[(1 - \frac{\lambda_e}{c_e \mu_e}) \sum_{n=0}^{c_e - 1} \frac{(\lambda_e)^n}{n!(\mu_e)^n} + \frac{(\lambda_e)^{c_e}}{c_e!(\mu_e)^{c_e}} \Big]^{-1} \cdot (c_e \mu_e - \lambda_e)^{-1} - \delta_k^{edge}$$

In order to keep the waiting queue in EC e is stable, we introduce a new constraint here:

$$c_e \cdot \mu_e \ge \sum_{k\in\mathcal{K}} x_{k,e}, \quad \forall e \in \mathcal{E} \tag{28}$$

According to [17], $f(\lambda_e) = 0$ can be numerically solved and finds the specific solution. Since $f(\lambda_e)$ is a c_e order polynomial, it may have c_e solutions in $f(\lambda_e) = 0$. We choose the upper bound following the **Algorithm** 1.

This yields that (13) is equivalent to:

$$\sum_{k\in K} x_{k,e} \le \lambda_e^{max}, \forall e \in \mathcal{E} \tag{29}$$

where λ_e^{max} is the output of **Algorithm** 1. Noting that, one of the inputs of **Algorithm**1 is the delay tolerance for edge cloud access δ_k^{edge}, while we have the total latency upper bound δ_k, which is the addition of edge cloud delay δ_k^{edge} and link delay δ_k^{link} in (8). Now the problem becomes how to determine a suitable proportion of δ_k^{link} and δ_k^{edge} given

Algorithm 1 Choosing λ_e^{max} from possible solutions

Input:
The number of VMs in edge cloud e, c_e;
The service rate for each VM in edge cloud e, μ_e;
The delay tolerance for edge cloud access, δ_k^{edge}
Output:
λ_e^{max};

Construct set $\Lambda = \{\lambda_i | f(\lambda_i) = 0, 0 < \lambda_i < c_e\mu_e\}$
Initialize $\lambda_e^{max} = \lambda_1$
for each λ_i, $\lambda_{i+1} \in \Lambda$ **do**
  Calculate $F = f(\frac{\lambda_i + \lambda_{i+1}}{2})$
  if $F < 0$ **then**
    $\lambda_e^{max} = \lambda_{i+1}$
  else
    break
  end if
end for

a fixed δ_k. To address this problem, we greedily assign as many flows as possible to the shortest path between d and e such that t_k^{link} is maximized and we let $\delta_k^{link} = t_k^{link}$, then we assign the rest proportion to δ_k^{edge}.

By introducing new variables and constraints in previous subsections, we could linearize the optimization problem and formulate as a Mixed Integer Linear Programming (MILP) Model. Thereby, the PCDG model is

$$\min_{\substack{x_{k,e}, y_{k,l}, z_l, \chi_e, \phi_{k,e} \\ \pi_{k,d,e}, \omega_{k,l}, \psi_{k,l,d,e}}} \quad TC^{new} \tag{30}$$

s.t. (7a)∼(7f), (10a)∼(10c), (16a)∼(16b), (19a)∼(19c), (23), (25a)∼(25g), (26)∼(29)

$$z_l \geq 0, \phi_{k,e} > 0, \quad \forall l \in \mathcal{L}, k \in \mathcal{K}, e \in \mathcal{E} \tag{30a}$$

$$x_{k,e}, \pi_{k,d,e} \in \{0, 1\}, \quad \forall k \in \mathcal{K}, d \in \mathcal{D}, e \in \mathcal{E} \tag{30b}$$

C. Scale Free Heuristics

With the increment of the number of flows, it is quite time-consuming to solve PCDG model. Therefore, we propose a greedy algorithm called GRC. It is worth noting that, the PCDG can always find an optimal solution with delay guarantees, while GRC here may suffer from QoS penalty when not satisfying delay tolerance. Hereafter we introduce a piecewise function Q_k^{time} as such penalty to measure the delayed impact on QoS:

$$Q_k^{time} = \begin{cases} 0, & t_k \leq \delta_k \\ \eta \cdot (t_k - \delta_k), & \text{otherwise} \end{cases} \tag{31}$$

where η is the penalty factor for missing the delay constraint.

The GRC try to assign the caching content to the nearest EC depending on the maximum user equipment's moving probability $P_{k,d}$. If such EC does not have enough storage space, then GRC try to cache in the second nearest EC from the same access router d, instead of dropping the information. For the GRC scheme, the idea is to choose the caching host by considering the path cost and node capacity. More details are illustrated in **Algorithm** 2 below.

Algorithm 2 Greedy Caching (GRC)

Input:
Flow set $\mathcal{K}$, Parameters in Table II
Output:
the total cost TC

Construct set $L_k = \{k | \eta_k > \eta_{k+1}, \forall k \in \mathcal{K}\}$
for each flow $k \in L_k$ and node $d \in \mathcal{D}$ **do**
  Find the maximum $P_{k,d}$ and related d,$P_{k,d} \leftarrow 0$
  Construct list L_e from d by dijkstra algorithm
  for each e in the list L_e **do**
    if $S_k \leq W_e^{re}$ and $S_k \leq W_t^{re}$ **then**
      $x_{k,e} \leftarrow 1$, $\pi_{k,d,e} \leftarrow 1, d \in \{d | P_{k,d} > 0, \forall d \in \mathcal{D}\}$
      $W_t^{re} \leftarrow W_t^{re} - S_k$,$W_e^{re} \leftarrow W_e^{re} - S_k$
      break
    end if
  end for
end for
for each flow $k \in L_k$ **do**
  Calculate $C_k^{host}, C_k^{cache}, C_k^{miss}, Q_k^{time}$ using (4), (5), (6) and (31) respectively
end for
$TC \leftarrow \sum_{k \in K} \{\alpha C_k^{host} + \beta C_k^{cache} + \gamma C_k^{miss} + Q_k^{time}\}$

III. Numerical Investigations

In this section we demonstrate the performance of proposed PCDG with other methods. All results presented hereafter are averaged over one thousand Monte Carlo iterations. The simulation parameters that assumed in the investigations are presented in Table II.

TABLE II
Simulation Parameters Used in This Paper [18][19].

weight of cache host (α)	[0,1]
weight of path cost (β)	[0,1]
weight of miss cost (γ)	[0,1]
penalty factor for delay (η)	[0,1]
size of cache items (S_k)	[10,500]Mbit
hit miss cost (C_k^{miss})	[100,1000]
link weight	[1,100]
edge cloud remaining cache space (W_e^{re})	[8,16]Gbit
network total remaining cache space (W_t^{re})	100 Gbit
flow (request) rate (R_k)	[0.064,10]Mbps
number of VMs (c_e)	8
service rate for each VM (μ_e)	(0,4]
link capacity (C_l)	2Gbps
delay tolerance (δ_k)	[0.03,60] s
changing point of attachment probability ($P_{k,d}$)	[0,1]

Figure 1 compares the total cost of these techniques. As can be seen from the figure, the no-cache strategy has, as expected, the lowest performance from all schemes. In order to avoid losing information, the all-cache algorithm tries to cache contents in all possible ECs, which reduces the price of redirection C_k^{cache} since the mobile user could get the subscriptions from the nearest EC immediately after handover, however, it incurs significant memory cost C_k^{host} because every EC hosts a copy of user's subscription. GRC performs better than all-cache, and similarly as PCDG before

20 flows, then the performance cost increases from 18.71% to 37.70% (PCDG vs GRC). This is expected by the fact that GRC only considers the redirected cost C_k^{cache} and caching memory limitation. But it has the risk to be punished by Q_k^{time} because of not satisfying time limitation, when GRC assigns a content to a node which has enough cache space but already maintains a long request waiting queue. Clearly, with the increase of the number of flows, the PCDG outperforms than GRC(albeit with higher computational complexity cost).

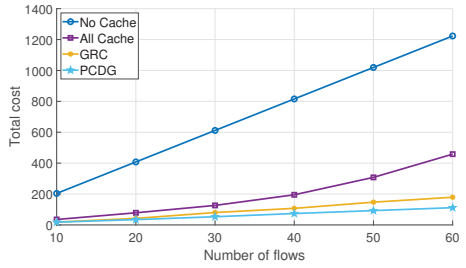

Fig. 1. Total cost with different number of flows.

Figure 2 illustrates the satisfied probability with different number of flows. With satisfied probability we denote the ratio of the number of flows that fulfill their latency constraint to the total number of flows. Without loss of generality, we keep the diversity for user requests, i.e. a partial of the requests is time-sensitive. In that respect, the proportion of time-nonsensitive task is set to be 50% (i.e. the satisfied probability of no-cache is kept at 50%) in this experiment. The all-cache and PCDG could assign all user requests flows for popular content to edge clouds that they satisfy their delay constraint; and this is the reason why the satisfied probability in all cases one in figure 2 for the two schemes. Whereas, the GRC suffers from outage which is increasing as the number of flows in the network is increasing. It is worth noticing that though the satisfied probability of all-cache is one, the total cost of all-cache is larger than GRC in figure 1 since all-cache contains a significant hosting cost.

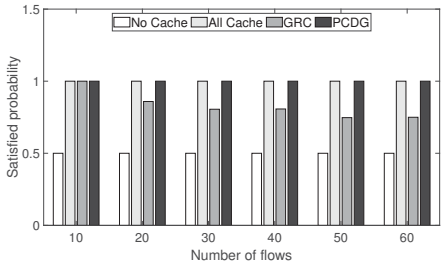

Fig. 2. Satisfied probability with different number of flows.

IV. Conclusions

Providing advanced caching strategies that incorporate delay constraints is of paramount importance to support emerging and future applications that require strict latency deadlines. In order to explicitly consider delay constraints, a Proactive Caching with Delay Guarantees (PCDG) strategy for mobile networks is proposed. To this end, a mathematical programming formulation is presented that amalgamates via suitable linearization queuing theory models to capture delays on both edge-clouds and in the network links. In addition a greedy algorithm is presented and with a wide set of numerical investigations the performance is evaluated.

References

[1] V. Cisco, "Cisco visual networking index: Forecast and methodology 2016–2021.(2017)," 2017.

[2] C. Fang, H. Yao, Z. Wang, W. Wu, X. Jin, and F. R. Yu, "A survey of mobile information-centric networking: Research issues and challenges," *IEEE Communications Surveys & Tutorials*, 2018.

[3] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[4] X. Vasilakos, V. A. Siris, G. C. Polyzos, and M. Pomonis, "Proactive selective neighbor caching for enhancing mobility support in information-centric networks," in *Proceedings of the second edition of the ICN workshop on Information-centric networking*, pp. 61–66, ACM, 2012.

[5] H. Farahat, R. Atawia, and H. S. Hassanein, "Robust proactive mobility management in named data networking under erroneous content prediction," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–6, IEEE, 2017.

[6] G. Zheng and V. Friderikos, "Optimal proactive cache management in mobile networks," in *Communications (ICC), 2016 IEEE International Conference on*, pp. 1–6, IEEE, 2016.

[7] V. Sourlas, G. S. Paschos, P. Flegkas, and L. Tassiulas, "Mobility support through caching in content-based publish/subscribe networks," in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, pp. 715–720, IEEE, 2010.

[8] U. Farooq, E. W. Parsons, and S. Majumdar, "Performance of publish/subscribe middleware in mobile wireless networks," in *ACM SIGSOFT Software Engineering Notes*, vol. 29, pp. 278–289, ACM, 2004.

[9] A. Gaddah and T. Kunz, "Extending mobility to publish/subscribe systems using a pro-active caching approach," *Mobile Information Systems*, vol. 6, no. 4, pp. 293–324, 2010.

[10] T. Hou, G. Feng, S. Qin, and W. Jiang, "Proactive content caching by exploiting transfer learning for mobile edge computing," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pp. 1–6, IEEE, 2017.

[11] S. Müller, O. Atan, M. van der Schaar, and A. Klein, "Context-aware proactive content caching with service differentiation in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 2, pp. 1024–1036, 2017.

[12] L. Rui, S. Yang, and H. Huang, "A proactive multi-level cache selection scheme to enhance consumer mobility support in named data networking," *International Journal of Distributed Sensor Networks*, vol. 13, no. 4, p. 1550147717700897, 2017.

[13] J. F. Kurose and K. W. Ross, *Computer networking: a top-down approach*. Addison-Wesley Reading, 2010.

[14] D. Bertsekas and R. Gallager, "Data networks. 1992," *PrenticeHall, Englewood Cliffs, NJ*, 1992.

[15] P. J. Burke, "The output of a queuing system," *Operations research*, vol. 4, no. 6, pp. 699–704, 1956.

[16] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control," *IEEE Transactions on Computers*, vol. 66, no. 5, pp. 810–819, 2017.

[17] V. Marianov and D. Serra, "Location models for airline hubs behaving as m/d/c queues," *Computers & Operations Research*, vol. 30, no. 7, pp. 983–1003, 2003.

[18] G. Zheng, C.-Y. Wang, V. Friderikos, and M. Dohler, "High mobility multi modal e-health services," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–7, IEEE, 2018.

[19] G. Zheng, A. Tsiopoulos, and V. Friderikos, "Optimal vnf chains management for proactive caching," *IEEE Transactions on Wireless Communications*, 2018.