Universally Stable Cache Networks

Yuanyuan Li, Stratis Ioannidis
E.C.E., Northeastern University, Boston, USA
{yuanyuanli,ioannidis}@ece.neu.edu

***Abstract*—We consider a cache network in which intermediate nodes equipped with caches can serve content requests. We model this network as a universally stable queuing system, in which packets carrying identical responses are consolidated before being forwarded downstream. We refer to resulting queues as $\mathtt{M/M/1c}$ or *counting queues*, as consolidated packets carry a counter indicating the packet's multiplicity. Cache networks comprising such queues are hard to analyze; we propose two approximations: one via $\mathtt{M/M/\infty}$ queues, and one based on $\mathtt{M/M/1c}$ queues under the assumption of Poisson arrivals. We show that, in both cases, the problem of jointly determining (a) content placements and (b) service rates admits a poly-time, $1-1/e$ approximation algorithm. Numerical evaluations indicate that both approximations yield good solutions in practice, significantly outperforming competitors.**

***Index Terms*—DR-submodularity, cache networks, Jackson networks**

I. Introduction

We consider a network of caches, in which intermediate nodes store requested contents and can serve content requests. Cache networks are a natural abstraction for many applications, including information-centric networks [1]–[3], content delivery networks [4]–[6], and peer-to-peer networks [7], [8]. A series of recent efforts focus on the problem of *cache network design*, describing algorithms for placing contents in caches in order to minimize routing costs [9]–[13].

In most prior work, routing costs are modeled via a linear function of traffic in each edge, which does not capture, e.g., delays due to queuing. A recent paper by Mahdian et al. [13] addresses this limitation by considering so-called *Kelly cache networks*, i.e., cache networks whose links are associated with $\mathtt{M/M/1}$ queues. This formulation has several advantages. First, it allows the authors to capture queuing costs in their cache network design. Second, the system is a Kelly network [14] and, hence, its steady-state distribution is easy to characterize. Unfortunately, these modelling advantages come at the expense of realism. In $\mathtt{M/M/1}$ queues, packets carrying *the same content* are queued and served separately. This would not happen in a real network, in which packets carrying identical content would be used to serve multiple requests. As a side-effect of this modeling distortion, networks studied by Mahdian et al. can become unstable: queue sizes can grow to infinity, congested with packets containing identical content.

In this work, we address this problem by considering a new type of queue, which we refer to as a *counting queue*. When packets containing identical content arrive in such a queue, they merge, resulting in a single packet carrying the same content. The header of this packet contains a counter with the "cardinality" of merged packets it represents. Merged packets are forwarded towards the request source, serving multiple requests via a single response. Counting queues, which we denote by $\mathtt{M/M/1c}$, capture real-life behavior more accurately than $\mathtt{M/M/1}$ queues. They also lead to networks that are *universally stable*: the merging of packets prevents queues (and counters) to grow to infinity, irrespectively of demand.

Nevertheless, by introducing $\mathtt{M/M/1c}$ queues, we suffer a reversal of fortune in comparison to Mahdian et al. [13]: though we gain realism, we lose tractability, as the resulting system is *not* a Kelly network, and steady-state distributions are hard to describe. As a result, their steady-state behavior, and the routing cost optimization that they correspond to, are difficult to characterize in a closed form. One of the main contributions of our work is to address this challenge directly, providing both analytical and experimental evidence that $\mathtt{M/M/1c}$ queues are well-approximated by $\mathtt{M/M/\infty}$ queues; the latter are indeed easy to analyze, enabling us to produce algorithms for the cache design problem with approximation guarantees. In particular, we make the following contributions:

- We introduce networks of $\mathtt{M/M/1c}$ queues, aiming to capture network behavior with greater realism. In contrast to $\mathtt{M/M/1}$ queues, resulting networks are *not* Kelly networks, and intermediate queue arrivals are *not* Poisson.
- We show that $\mathtt{M/M/\infty}$ queues approximate $\mathtt{M/M/1c}$ queues; we show this both experimentally and analytically, through a mutual stochastic dominance (c.f. Thm. 1). Most importantly, both queues lead to networks that are universally stable.
- Motivated by the above observations, we study two cache network design problems, each serving as an approximation of a cache network with counting queues. Both problems optimize content placement and service assignment decisions *jointly*. In the first problem, $\textsc{MinCost}_{\mathtt{M/M/\infty}}$, we approximate counting queues with $\mathtt{M/M/\infty}$ queues; in the second problem, $\textsc{MinCost}_{\mathtt{M/M/1c}}$, we use $\mathtt{M/M/1c}$ steady-state distributions, assuming however Poisson arrivals in intermediate queues.
- We show that both problems are NP-hard (c.f. Thm. 2), and construct a $1-1/e$ poly-time approximation algorithm for the joint optimization of item placements and service assignments (c.f. Thm. 4).
- Finally, we conduct extensive experiments over multiple topologies: our joint item placements and service rate

The authors gratefully acknowledge support from National Science Foundation grants NeTS-1718355 and CCF-1750539.

assignments significantly outperform competitors.

From a technical standpoint, our algorithm solves a mixed integer problem with a non-convex objective; this requires showing that both $\text{MINCOST}_{\texttt{M/M/}\infty}$ and $\text{MINCOST}_{\texttt{M/M/1c}}$ exhibit an important underlying structural property (c.f. Theorem 3): their objectives are continuous *Diminishing Returns (DR) submodular* [15] w.r.t. both content placements *and* service assignments *jointly*, a result that is non-obvious.

The remainder of this paper is structured as follows. We review related work in Sec. II. We introduce M/M/1c and M/M/∞ queues, and formulate our two approximate problems in Sec. III. Section IV contains our approximation algorithms. Our experiments are in Sec. V, and we conclude in Sec. VI.

II. Related Work

Cache networks have been intensely studied both experimentally and theoretically. Several works [16]–[22] model the network as a bipartite graph, in which requests fetch contents in one hop, and proposed algorithms do not readily generalize to arbitrary topologies. Multi-hop networks are studied by a series of recent papers [9]–[12], [23], all of which assume costs are linear functions of traffic. As such, they cannot be used to model costs in queuing systems like the ones we study.

Dehgan et al. [21] and Ioannidis and Yeh [11] consider the joint optimization of caching and routing in networks; Dehghan et al. in particular study routing in the bipartite setting, while Ioannidis and Yeh [11] do so in arbitrary topologies. Our joint optimization of caching and service rates is fundamentally different, not only because it contains both continuous and integer variables; it is also not amenable to standard submodularity approaches, as is [11], but requires the use of continuous DR-submodular optimization [15] instead. Zafari et al. [24] jointly optimize data compression rate and data placement in a tree topology, posing this as a mixed integer problem; they solve this by a spatial branch-and-bound search strategy, which comes with no poly-time guarantees.

Maximizing a submodular function subject to a matroid constraint is classic. Krause and Golovin [25] show that the greedy algorithm achieves a $1/2$ approximation ratio. Calinescu et al. [26] propose a *continuous greedy* algorithm improving the ratio to $1 - 1/e$, that applies a Frank-Wolfe [27] variant to the multilinear extension of the submodular objective. Further improvements are made by Sviridenko et al. [28] for a more restricted class of submodular functions. Bian et al. [29] [15] show that the same Frank-Wolfe variant can be used to maximize continuous DR-submodular functions within a $1 - 1/e$ ratio. One of our technical contributions is to show that the multilinear extension, in our case, which is a function of both randomized item placements *and* continuous service rates, is jointly DR-submodular in its input. We note that we depart from multilinear extensions considered in prior work [13], [26], [28], [30], that did not contain continuous variables beyond the ones due to randomization.

Our work is closest to, and inspired by, recent work Mahdian et al. [13]. As discussed in the introduction, they consider a cache network in which each edge is associated with an M/M/1 queue. Resulting costs are not linear, capture queuing, and the objective is submodular and therefore optimizable via the continuous greedy algorithm of Calinescu et al. [26]. We again depart by considering M/M/1c (counting) queues and M/M/∞ approximations thereof, and optimizing item placement and routing decisions *jointly*: as a result, our optimization requires tools beyond classic submodularity.

III. Problem Formulation

We consider a network of caches which store a finite number of items. Requests for items are generated and are routed through pre-determined paths. Upon hitting a cache which stores the requested item, a response carrying the item is back-propagated over the reverse path. This generates traffic over queues on network edges. We aim to minimize traffic costs by (a) placing items in caches and (b) assigning queue service rates across responses appropriately. In what follows, we describe this problem in detail.

A. System Model

Following Mahdian et al. [13], we consider a network modeled as a directed graph $G(V, E)$ with node set V. Each edge e in the graph is represented by $e = (u, v) \in E$, where $u, v \in V$. This directed graph is symmetric, i.e., if $(u, v) \in E$, then $(v, u) \in E$ as well.

1) Caches: Items of equal size are permanently stored in certain network nodes, called *designated servers*. Formally, for every item $i \in \mathcal{C}$, where set $\mathcal{C}$ is the *item catalog*, we denote by $\mathcal{S}_i \subseteq V$ the set of designated servers storing i. Every node in V, including designated servers, has additional storage that is used to store more items from the catalog. Formally, each node v is associated with a cache of finite storage capacity $c_v \in \mathbb{N}$. We use a binary variable $x_{vi} \in \{0, 1\}$ indicating whether node $v \in V$ is caching item $i \in \mathcal{C}$. Let vector $\boldsymbol{x} = [x_{vi}]_{v \in V, i \in \mathcal{C}} \in \{0, 1\}^{|V||\mathcal{C}|}$ be the global item placement vector. We denote the set of feasible placements by:

$$\mathcal{D} = \{\boldsymbol{x} \in \{0,1\}^{|V||\mathcal{C}|} : \textstyle\sum_{i \in \mathcal{C}} x_{vi} \leq c_v, \forall v \in V\}. \quad (1)$$

2) Requests and Responses: A set of nodes $\mathcal{Q} \subseteq V$, called *query nodes*, generate requests to fetch items from $\mathcal{C}$. Let $\mathcal{R}$ be the set of request types. Each request has a unique type $r \in \mathcal{R}$, determined by (a) the item $i^r \in \mathcal{C}$ being requested, and (b) the *path* $p^r \subseteq V$ followed by the request. We make the following assumptions on path $p^r, r \in \mathcal{R}$: (a) p^r is a sequence of adjacent nodes, e.g. p_1^r, p_2^r, ..., p_K^r, where $(p_i^r, p_{i+1}^r) \in E$, (b) $p_1^r \in \mathcal{Q}$, i.e., the first node of path is a query node, (c) $p_K^r \in \mathcal{S}_{i^r}$, i.e., the last node of path is a designated server, and (d) the path p^r is simple, i.e., it does not contain repeated nodes. For $v \in p^r$, let $k_{p^r}(v) \in \{1, 2, ..., K\}$ be the position of node v in path p^r, i.e., $k_{p^r}(v) = k$ iff $p_k^r = v$.

Requests of type r are generated according to an exogenous Poisson process with rate $\lambda^r \geq 0, r \in \mathcal{R}$. Then, they follow path p^r; when the request reaches a node storing item i^r, a response is generated. This response carries item i^r to query node $p_1^r \in \mathcal{Q}$ following the reverse path. Given all the paths $\{p^r, r \in \mathcal{R}\}$, for every edge $e \in E$, we denote by $\mathcal{R}_e$ the set of response types passing through edge e, i.e., for $e=(v,u)$, $\mathcal{R}_{(v,u)}=\{r\in\mathcal{R}:(u,v)\in p^r\}$.

3) Queues and Costs: We assume requests are negligible, but responses incur traffic in the network. We model this traffic as follows. Every edge $e\in E$ is associated with service rate $\mu_e\in\mathbb{R}_+$. The service rate in an edge is split across response types. For every type $r\in\mathcal{R}_e$, there exists a queue with service rate μ_e^r. Assume that the minimum service rate for all queues is some small $\epsilon\in\mathbb{R}_+$. Let vector $\boldsymbol{\mu}=[\mu_e^r]_{e\in E,r\in\mathcal{R}_e}\in\mathbb{R}_+^{\sum_e|\mathcal{R}_e|}$ be the global service rate assignment vector. We denote the set of feasible assignments by:

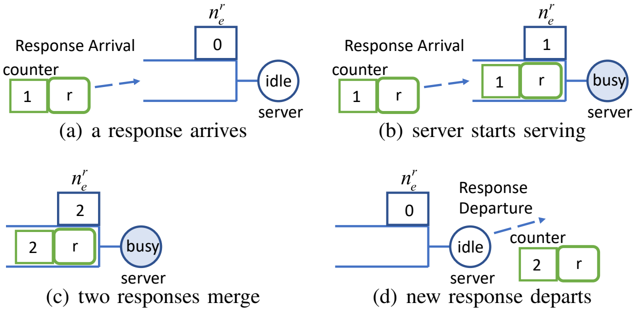
$$\mathcal{D}_\mu=\{\boldsymbol{\mu}\in\mathbb{R}_+^{\sum_e|\mathcal{R}_e|}: \mu_e^r\geq\epsilon, \sum_{r\in\mathcal{R}_e}\mu_e^r\leq\mu_e, \forall e\in E, r\in\mathcal{R}_e\}. \quad (2)$$

Let $n_e^r\in\mathbb{N}$ be the queue size. We assume that traffic cost is a function of n_e^r and denote by $c_e^r(n_e^r):\mathbb{N}\to\mathbb{R}_+$ the cost of response type r on edge e. The global service rate assignment $\boldsymbol{\mu}$ and the global item placement $\boldsymbol{x}$ are design parameters: we wish to determine $\boldsymbol{x}$ and $\boldsymbol{\mu}$ jointly to minimize expected traffic costs in steady state. Before we state this optimization formally, we first describe the queues we consider.

B. Queue Types

Mahdian et al. [13] consider M/M/1 queues: all responses are served individually by the edge server. This is convenient from a modeling perspective, because M/M/1 queues form a Kelly network [14]. However, transmitting same-type responses individually over the same queue is both inefficient and impractical. If two responses of the same type are present in a queue, it suffices to transmit only one of them: requests pending at the same downstream source can be satisfied simultaneously by the same response. Transmitting responses individually leads to larger queue sizes, thereby incurring larger traffic costs, but also larger delays (by Little's Theorem [31]). In fact, the system considered by Mahdian et al. becomes unstable when the load of an M/M/1 queue is above one. This motivates us to introduce a new type of queue we call a *counting queue*. As we discuss below, this comes at the cost of increasing model complexity: the network resulting from counting queues, albeit more realistic (and, most importantly, universally stable), is *not* a Kelly network and is, hence, harder to analyze.

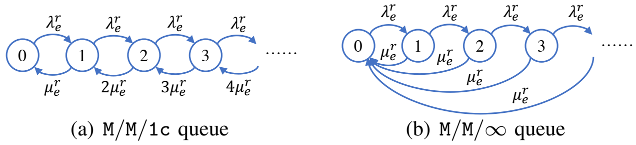
1) M/M/1c Queue: To model realistic behavior, a *counting queue* behaves as follows. When a response of type r arrives at an empty queue on edge e, it experiences immediate service with rate μ_e^r. A subsequent response of type r arriving before the server is finished is *not* queued: instead, it merges with the response in the server, and both are served simultaneously with rate μ_e^r. In practice, this is implemented as follows: every response is associated with a counter initialized to one by the designated server generating it. Whenever two responses of type r are collocated in an edge e, they merge into a new response of type r, with a counter equal to the sum of its constituent counters. Note that, as service times are exponential, by the memoryless property [32], the residual service time after a merge remains exponential with rate μ_e^r. After being served, this merged response departs. This whole process is depicted in Fig. 1. We formally refer to such a queue as an M/M/1c queue ('c' is for 'counter'). We consider the size n_e^r of an M/M/1c queue to be equal to the counter value of the merged response in the queue's server. Assuming Poisson arrivals of responses with counters equal to one, the queue size process $\{n_e^r(t);t\geq 0\}$ is a Markov process whose transition diagram is described in Fig. 2(a), and its steady state distribution is given by the following lemma:



(a) a response arrives (b) server starts serving

(c) two responses merge (d) new response departs

Fig. 1: M/M/1c queue. When two responses meet in a queue, they merge as a new response with counter value equal to sum of their respective counters. Queue size n_e^r equals the counter value of the packets in this queue, and 0 if the queue is empty.



(a) M/M/1c queue (b) M/M/∞ queue

Fig. 2: M/M/1c and M/M/∞ transitions, assuming Poisson arrivals.

Lemma 1. *Assume that response arrivals follow a Poisson process with rate λ_e^r, and each response's counter is 1. Then, the steady state distribution of the M/M/1c queue is $\boldsymbol{P}_{\mathtt{M/M/1c}}(n_e^r=n)=(\frac{\rho_e^r}{\rho_e^r+1})^n(\frac{1}{\rho_e^r+1})$. where $\rho_e^r=\frac{\lambda_e^r}{\mu_e^r}$ is load of response type r on edge e.*

Note that this queue is *universally stable*, i.e., positive recurrent for all $\rho_e^r>0$. However, Lemma 1 only holds for edges directly adjacent to a designated server. This is because intermediate queues, further from a designated server, satisfy neither of the two assumptions of Lemma 1: (a) arrivals are *not* Poisson, and (b) counters of responses may be *larger than 1*. Overall, the entire system is *not* a Kelly network, and its steady state distribution is difficult to describe in a closed form.

2) M/M/∞ Queue: The above state of affairs motivates us to approximate counting queues with M/M/∞ queues [33]. Recall that an M/M/∞ queue has infinite servers. Just as in an M/M/1c queue, incoming responses are not queued but are immediately served with service rate μ_e^r. However, the service times of responses collocated in an M/M/∞ queue are independent, while in M/M/1c they are tightly coupled: in fact, all responses are served simultaneously. Again, $\{n_e^r(t);t\geq 0\}$ is a Markov process whose transition diagram is described in Fig. 2(b), and its steady state distribution is given by the following lemma:

TABLE I: Notation Summary

$\mathbb{R}, \mathbb{N}$	Sets of real and natural numbers
$\mathbb{R}_+, \mathbb{N}_+$	Sets of non-negative reals and positive naturals
$G(V, E)$	Network graph, with nodes V and edges E
$k_{p^r}(v)$	Position of node v in path p^r
c_v	Cache capacity at node $c \in V$
x_{vi}	Integer variable indicating $v \in V$ stores $i \in \mathcal{C}$
$\boldsymbol{x}$	Global item placement vector of x_{vi}s in $\{0,1\}^{\lvert V \rvert \lvert \mathcal{C} \rvert}$
$\mathcal{R}$	Set of types of requests
$\mathcal{R}_e$	Set of types of responses passing through e
λ_r	Request arrival rate for type $r \in \mathcal{R}$
μ_e	Service rate of edge $e \in E$
μ_e^r	Service rate of type $r \in \mathcal{R}_e$ over edge e
ϵ	The minimum service rate of all μ_e^r
$\boldsymbol{\mu}$	Global service rates vector of μ_e^rs in $\mathbb{R}_+^{\sum_e \lvert \mathcal{R}_e \rvert}$
ρ_e^r	Load of type r over edge e
$\mathcal{D}$	Set of feasible item placements $\boldsymbol{x}$
$\mathcal{D}_\mu$	Set of feasible service rates $\boldsymbol{\mu}$
$C_{\texttt{M/M/}\infty}$	Cost function for $\texttt{M/M/}\infty$ queues
$C_{\texttt{M/M/1c}}$	Cost function for $\texttt{M/M/1c}$ queues
C	Generalized non-decreasing and convex cost function
F	Caching gain of decision $\{\boldsymbol{x}, \boldsymbol{\mu}\}$ over $\{\mathbf{0}, \boldsymbol{\epsilon}\}$
$\tilde{\mathcal{D}}$	Convex hull of $\mathcal{D}$
y_{vi}	Probability that v stores i
$\boldsymbol{y}$	Vector of marginal probabilities $y_v i$s in $[0,1]^{\lvert V \rvert \lvert \mathcal{C} \rvert}$
G	Multilinear extension with marginals $\boldsymbol{y}$
$[\boldsymbol{x}]_{+(v,i)}$	Vector $\boldsymbol{x}$ with the (v,i)-th coordinate set to 1
$[\boldsymbol{x}]_{-(v,i)}$	Vector $\boldsymbol{x}$ with the (v,i)-th coordinate set to 0

Lemma 2. *Assume that response arrivals follow a Poisson process with rate λ_e^r. Then, the steady state distribution of* $\texttt{M/M/}\infty$ *queue is* $\boldsymbol{P}_{\texttt{M/M/}\infty}(n_e^r = n) = \frac{(\rho_e^r)^n}{n!} e^{-\rho_e^r}$, *where* $\rho_e^r = \frac{\lambda_e^r}{\mu_e^r}$ *is load of response type r on edge e.*

Note that responses here do not merge and, hence, implicitly all have counter value one. There are several reasons why $\texttt{M/M/}\infty$ queues are good approximations of $\texttt{M/M/1c}$ queues. First, observe that both queues are universally stable. Second, they exhibit the same aggregate service rate: when n_e^r customers are in the queue, the aggregate service rate in both is $n_e^r \mu_e^r$; put differently, in both queues the aggregate service rate grows linearly with the queue size. Finally, queue sizes of $\texttt{M/M/1c}$ and $\texttt{M/M/}\infty$ queues are related through a notion of mutual stochastic dominance. In particular, it is easy to confirm from Lemmas 1 and 2 that the two queues have the same expectation, i.e.,

$$\mathbb{E}_{\texttt{M/M/1c}}[n_e^r] = \mathbb{E}_{\texttt{M/M/}\infty}[n_e^r] = \rho_e^r. \tag{3}$$

More generally, all moments of the two queues are coupled through the following relationship:

Theorem 1. *Let* $m^k_{\texttt{M/M/1c}}(\rho_e^r) = \mathbb{E}_{\texttt{M/M/1c}}[(n_e^r)^k]$ *and* $m^k_{\texttt{M/M/}\infty}(\rho_e^r) = \mathbb{E}_{\texttt{M/M/}\infty}[(n_e^r)^k]$ *be the k-th moment of n_e^r in* $\texttt{M/M/1c}$ *and* $\texttt{M/M/}\infty$ *queues, respectively. Then, for all $\rho_e^r \geq 0$,*

$$m^k_{\texttt{M/M/}\infty}(\rho_e^r) \leq m^k_{\texttt{M/M/1c}}(\rho_e^r) \leq k! \cdot m^k_{\texttt{M/M/}\infty}(\rho_e^r). \tag{4}$$

The proof can be found in Appendix B. This theorem immediately implies that, for any polynomial cost function $c_e^r(n_e^r)$, the expected costs under the two queues are within a multiplicative constant (not depending on ρ_e^r) of each other.[1]

A significant advantage of $\texttt{M/M/}\infty$ queues is that they are reversible [32]. Hence, networks of such queues form a Jackson network [35]. In steady state, departures from these queues are Poisson by Burke's theorem [36], and steady state distributions can be characterized (c.f. Lemma 4 in Appendix A).

C. Cache Cost Minimization

As discussed above, given item placements $\boldsymbol{x} \in \mathcal{D}$ and service rate assignments $\boldsymbol{\mu} \in \mathcal{D}_\mu$, the network of $\texttt{M/M/}\infty$ queues is a Jackson network. Arrivals of responses of type r on edge $e = (v, u)$ where $(u, v) \in p^r$ are Poisson with rate:

$$\lambda_e^r = \lambda_e^r(\boldsymbol{x}) = \lambda^r \textstyle\prod_{k'=1}^{k_{p^r}(u)} (1 - x_{p^r_{k'}, i^r}). \tag{5}$$

Intuitively, this states that responses of type r pass through edge $(v, u) \in E$ iff all path predecessors of node v do not store item i^r, i.e., $x_{v' i^r} = 0$ for all $v' : k_{p^r}(v') < k_{p^r}(v)$. With Poisson arrivals, the expected cost of response type $r \in \mathcal{R}_e$ on edge $e \in E$, according to Lem. 2, is:

$$\mathbb{E}_{\texttt{M/M/}\infty}[c_e^r(n_e^r)] = \textstyle\sum_{n=0}^{\infty} c_e^r(n) \cdot e^{-\rho_e^r} \frac{(\rho_e^r)^n}{n!}, \tag{6}$$

where the load of response type $r \in \mathcal{R}_e$ on $e = (u, v) \in E$ is:

$$\rho_e^r = \rho_e^r(\boldsymbol{x}, \mu_e^r) = \tfrac{\lambda^r}{\mu_e^r} \textstyle\prod_{k'=1}^{k_{p^r}(v)} (1 - x_{p^r_{k'}, i^r}). \tag{7}$$

Given a cache network by graph $G(V, E)$, service rate capacities μ_e, $e \in E$, storage capacities c_v, $v \in V$, a requests set $\mathcal{R}$ and arrival rates λ_r, $r \in \mathcal{R}$, we formulate the cache cost minimization problem under $\texttt{M/M/}\infty$ queues as follows:

$$\textsc{MinCost}_{\texttt{M/M/}\infty}$$

$$\min_{\boldsymbol{x}, \boldsymbol{\mu}} : C_{\texttt{M/M/}\infty}(\boldsymbol{x}, \boldsymbol{\mu}) = \sum_{e \in E} \sum_{r \in \mathcal{R}_e} \mathbb{E}_{\texttt{M/M/}\infty}[c_e^r(n_e^r)], \tag{8a}$$

$$\text{s.t.} : \boldsymbol{x} \in \mathcal{D}, \ \boldsymbol{\mu} \in \mathcal{D}_\mu, \tag{8b}$$

where $\mathcal{D}$ is defined by (1) and $\mathcal{D}_\mu$ is defined by (2).

Similarly, if the assumptions of Lemma 1 hold, the expected cost of an $\texttt{M/M/1c}$ queue is:

$$\mathbb{E}_{\texttt{M/M/1c}}[c_e^r(n_e^r)] = \textstyle\sum_{n=0}^{\infty} c_e^r(n) \cdot (\frac{\rho_e^r}{\rho_e^r + 1})^n (\frac{1}{\rho_e^r + 1}). \tag{9}$$

where ρ_e^r are given by (7). Based on this observation, we also consider a cache cost minimization problem under $\texttt{M/M/1c}$ queues, defined as:

$$\textsc{MinCost}_{\texttt{M/M/1c}}$$

$$\min_{\boldsymbol{x}, \boldsymbol{\mu}} : C_{\texttt{M/M/1c}}(\boldsymbol{x}, \boldsymbol{\mu}) = \sum_{e \in E} \sum_{r \in \mathcal{R}_e} \mathbb{E}_{\texttt{M/M/1c}}[c_e^r(n_e^r)], \tag{10a}$$

$$\text{s.t.} : \boldsymbol{x} \in \mathcal{D}, \ \boldsymbol{\mu} \in \mathcal{D}_\mu, \tag{10b}$$

We stress that *both* problems (8) and (10) are approximations of networks of counting queues. $\textsc{MinCost}_{\texttt{M/M/}\infty}$ is clearly an approximation as $\texttt{M/M/}\infty$ queues are used instead of $\texttt{M/M/1c}$ queues. The objective (8a) captures steady state costs in such a system accurately, as arrivals in intermediate queues are indeed Poisson. $\textsc{MinCost}_{\texttt{M/M/1c}}$ is an approximation as the objective assumes Poisson arrivals and counters of size 1 at intermediate queues, neither of which are true for a real network of $\texttt{M/M/1c}$ queues. As we see in Sec. V, these approximations appear to perform well experimentally. Nevertheless, both problems are hard; we prove the following in Appendix C:

Theorem 2. *Problems* (8) *and* (10) *are NP-hard.*

[1] This result can be extended to continuous functions using, e.g., the Stone-Weierstrass Theorem [34].

IV. Main Results

In this section, we show how to solve Problems (8) and (10) within a constant approximation, poly-time algorithm. Mahdian et al. [13] approach caching problems via submodular maximization. However, (8) and (10) can not be cast in this setting, as they have mixed constraints: we would like to determine not only item placements (integer variables) but also service rates (continuous variables). Nevertheless, we construct a $1-1/e$-approximation poly-time algorithm. A crucial step is that the so-called multilinear extensions of (8a) and (10a) are *jointly* DR-submodular [29] w.r.t. $\boldsymbol{x}$ and $\boldsymbol{\mu}$.

A. Cache Gain Maximization

We introduce the following assumption on cost functions:

Assumption 1. *For all $r \in \mathcal{R}$, $e \in E$, and $n \in \mathbb{N}_+$,*

$$c_e^r(n+1) - c_e^r(n) \geq c_e^r(n) - c_e^r(n-1) \geq 0.$$

Using this assumption, we establish the following property:

Lemma 3. *Under Assumption 1, the expected cost functions $\mathbb{E}_{\texttt{M/M/}\infty}[c_e^r(n_e^r)]$ and $\mathbb{E}_{\texttt{M/M/1c}}[c_e^r(n_e^r)]$, given by* (6) *and* (9)*, are non-decreasing and convex w.r.t. load ρ_e^r, given by* (7).

The proof is in Appendix D. Motivated by Lemma 3, we consider a more general class of problems of the form:

MinCost

$$\min_{\boldsymbol{x},\boldsymbol{\mu}} : C(\boldsymbol{x},\boldsymbol{\mu}) = \sum_{e\in E}\sum_{r\in\mathcal{R}_e} C_e^r(\rho_e^r(\boldsymbol{x},\mu_e^r)), \tag{11a}$$

$$\text{s.t.} : \boldsymbol{x} \in \mathcal{D},\ \boldsymbol{\mu} \in \mathcal{D}_\mu, \tag{11b}$$

where expected cost functions $C_e^r : \mathcal{D} \times \mathcal{D}_\mu \to \mathbb{R}_+$ are non-decreasing and convex. Clearly, by Lem. 3, an algorithm solving (11) can also solve both (8) and (10). Following [10], [11], we consider an equivalent maximization problem instead:

MaxGain

$$\max_{\boldsymbol{x},\boldsymbol{\mu}} : F(\boldsymbol{x},\boldsymbol{\mu}) = C(\boldsymbol{0},\boldsymbol{\epsilon}) - C(\boldsymbol{x},\boldsymbol{\mu}), \tag{12a}$$

$$\text{s.t.} : \boldsymbol{x} \in \mathcal{D},\ \boldsymbol{\mu} \in \mathcal{D}_\mu, \tag{12b}$$

where $\boldsymbol{0} \in \mathcal{D}$ is the empty cache placement, $\boldsymbol{\epsilon} = \epsilon \cdot \boldsymbol{1} \in \mathcal{D}_\mu$ is the vector of minimum service rates, and $C(\boldsymbol{0},\boldsymbol{\epsilon})$ is an upper bound on $C(\boldsymbol{x},\boldsymbol{\mu})$. MinCost and MaxGain are equivalent, because (11a) and (12a) only differ by a constant $C(\boldsymbol{0},\boldsymbol{\epsilon})$.

B. DR-Submodularity

Let $\tilde{\mathcal{D}} = \text{conv}(\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{D}\}) \subseteq [0,1]^{|V||\mathcal{C}|}$ be the convex hull of the constraint set $\mathcal{D}$. That is:

$$\tilde{\mathcal{D}} = \{\boldsymbol{y} \in [0,1]^{|V||\mathcal{C}|} : \textstyle\sum_{i\in\mathcal{C}} y_{vi} \leq c_v, \forall v \in V\}. \tag{13}$$

Given a $\boldsymbol{y} \in \tilde{\mathcal{D}}$, consider a random vector $\boldsymbol{x}$ generated as follows: every $x_{vi} \in \{0,1\}$ is an independent Bernoulli variable such that $\mathbf{P}(x_{vi}=1) = y_{vi}$. The multilinear extension [26] $G(\boldsymbol{y},\boldsymbol{\mu}) : \tilde{\mathcal{D}} \times \mathcal{D}_\mu \to \mathcal{R}_+$ of F is $\mathbb{E}_y[F(\boldsymbol{x},\boldsymbol{\mu})]$, i.e.:

$$G(\boldsymbol{y},\boldsymbol{\mu}) = \sum_{\boldsymbol{x}\in\{0,1\}^{|V||\mathcal{C}|}} F(\boldsymbol{x},\boldsymbol{\mu}) \times \prod_{(v,i)\in V\times\mathcal{C}} y_{vi}^{x_{vi}}(1-y_{vi})^{1-x_{vi}}. \tag{14}$$

Given an $\mathcal{X} \subseteq \mathbb{R}^d$, we say that a function $f : \mathcal{X} \to \mathbb{R}$ is *DR-submodular* [15], if for all $\boldsymbol{a} \leq \boldsymbol{b} \in \mathcal{X}$, and all $i \in \mathbb{N}$, $k \in \mathbb{R}_+$, s.t. $(k\boldsymbol{e}_i + \boldsymbol{a})$ and $(k\boldsymbol{e}_i + \boldsymbol{b})$ are in $\mathcal{X}$, we have $f(k\boldsymbol{e}_i + \boldsymbol{a}) - f(\boldsymbol{a}) \geq f(k\boldsymbol{e}_i + \boldsymbol{b}) - f(\boldsymbol{b})$. The following lemma establishes that G is DR-submodular over the extended domain $\tilde{\mathcal{D}} \times \mathcal{D}_\mu$:

Theorem 3. *Under Assumption 1, the multilinear extension G is non-decreasing DR-submodular jointly on both $\boldsymbol{\mu}$ and $\boldsymbol{y}$.*

The proof is in Appendix E. This property is key; despite the fact that G is not concave, DR-submodularity implies we can maximize it within a constant factor. We stress here that the property holds jointly for $\boldsymbol{y}$ and $\boldsymbol{\mu}$, which is non-obvious.

C. Algorithm Overview

Leveraging Thm. 3, our algorithm consists of two steps:

Step 1: DR-submodular maximization. We first apply a variant of the Frank-Wolfe algorithm [29], summarized in Alg. 1, on the multilinear extension G. For brevity, we join $\boldsymbol{y}$ and $\boldsymbol{\mu}$ as one variable $\boldsymbol{z} = \{\boldsymbol{y},\boldsymbol{\mu}\} \in \mathcal{D}_z \equiv \{\boldsymbol{z} \in \tilde{\mathcal{D}} \times \mathcal{D}_\mu\}$. The algorithm first initializes the solution as $\boldsymbol{z} = \{\boldsymbol{0},\boldsymbol{\epsilon}\}$. Then, it iterates over the following steps:

$$\boldsymbol{m}_k \leftarrow \arg\max_{\boldsymbol{m}\in\mathcal{D}_z} \langle \boldsymbol{m}, \widehat{\nabla G}(\boldsymbol{z}_k)\rangle, \tag{15a}$$

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \gamma_k \boldsymbol{m}_k, \tag{15b}$$

where $\widehat{\nabla G}(\boldsymbol{z}_k)$ is an estimate of the gradient of G, and γ_k is an appropriately chosen step size. An estimator of ∇G is needed because both ∇G and G, given by (14), contain an exponential (in $|V||\mathcal{C}|$) number of terms. We describe this estimator in detail in Section IV-D. Given $\widehat{\nabla G}(\boldsymbol{z}_k)$, (15) is a linear program, which can be solved in polynomial time [29]. After K iterations, we get a fractional solution $\boldsymbol{z}_K = \{\boldsymbol{y}_K,\boldsymbol{\mu}_K\}$, i.e., the output of Alg. 1.

Step 2: Rounding. Finally, the fractional solution $\boldsymbol{y}_K$ is rounded into an integer solution $\boldsymbol{x}_K$. We describe how to do this in Sec. IV-E. This produces an approximate solution $\boldsymbol{x}_K$ and $\boldsymbol{\mu}_K$ to MaxGain.

Intuitively, DR-sumbodular maximization step (15) solves:

$$\max_{\boldsymbol{y},\boldsymbol{\mu}} : G(\boldsymbol{y},\boldsymbol{\mu}), \tag{16a}$$

$$s.t. : \boldsymbol{y} \in \tilde{\mathcal{D}},\ \boldsymbol{\mu} \in \mathcal{D}_\mu, \tag{16b}$$

where $\tilde{\mathcal{D}}$ is defined by (13). Alg. 1/Eq. (15) only solves (16) approximately because objective (16a) is not concave. Nevertheless, because (16a) is DR-submodular by Lemma 3, Alg. 1 produces a $1-1/e$ approximation to (16); see Lemma 8 in Appendix F for details. Combined with the rounding step, the following theorem characterizes the approximation guarantee of the overall algorithm:

Theorem 4. *Let $\boldsymbol{x}^*$, $\boldsymbol{\mu}^*$ be an optimal solution to* (12)*, $\boldsymbol{\mu}_K$ be the output of Frank-Wolfe variant, and $\boldsymbol{x}_K$ the integer solution after rounding. Then, with high probability,*

$$\mathbb{E}[F(\boldsymbol{x}_K,\boldsymbol{\mu}_K)] \geq (1-\tfrac{1}{e})F(\boldsymbol{x}^*,\boldsymbol{\mu}^*). \tag{17}$$

Algorithm 1: Frank-Wolfe variant for $G(\boldsymbol{z})$

```
Input: $G(\boldsymbol{z})$, $\mathcal{D}_z$, step size $\gamma \in (0,1]$, initial point $\{\mathbf{0}, \boldsymbol{\epsilon}\}$
. $t \leftarrow 0$, $k \leftarrow 0$, $\boldsymbol{z}_0 \leftarrow \{\mathbf{0}, \boldsymbol{\epsilon}\}$
while $t < 1$ do
    $\boldsymbol{m}_k \leftarrow \arg\max_{\boldsymbol{m} \in \mathcal{D}_z} \left\langle \boldsymbol{m}, \widehat{\nabla G}(\boldsymbol{z}_k) \right\rangle$
    $\gamma_k \leftarrow \min\{\gamma, 1 - t\}$
    $\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + \gamma_k \boldsymbol{m}_k$, $t \leftarrow t + \gamma_k$, $k \leftarrow k + 1$
end
return $\boldsymbol{z}_k$
```

The "with high probability" is w.r.t. the randomness of the estimator, while the expectation in (17) is w.r.t. the randomness in the rounding step. The proof is in Appendix F.

D. Estimator of Gradient

Eq. (15a) presumes access to the gradient ∇G. Nonetheless, both G and ∇G involve a summation over $2^{|V||\mathcal{C}|}$ terms. To create a poly-time algorithm, the usual approach is to use a sampling-based estimator [26]. In short, the partial derivatives of G w.r.t. y_{vi} and μ_e^r are (see [26] for (18)):

$$\tfrac{\partial G(\boldsymbol{x},\boldsymbol{\mu})}{\partial y_{vi}} = \mathbb{E}_y[C(\boldsymbol{x},\boldsymbol{\mu})|x_{vi}{=}0] - \mathbb{E}_y[C(\boldsymbol{x},\boldsymbol{\mu})|x_{vi}{=}1], \quad (18)$$

$$\tfrac{\partial G(\boldsymbol{x},\boldsymbol{\mu})}{\partial \mu_e^r} = \tfrac{1}{\mu_e^r}\mathbb{E}_y[\tfrac{\partial C_e^r(\rho_e^r)}{\partial \rho_e^r} \cdot \rho_e^r]. \quad (19)$$

One can thus estimate the gradient by (a) producing T random samples $\boldsymbol{x}^{(l)}$, $l = 1, ..., T$ of the random vector $\boldsymbol{x}$, consisting of independent Bernoulli coordinates with $\mathbf{P}(x_{vi} = 1) = y_{vi}$, and (b) computing the empirical mean w.r.t. y_{vi}:

$$\widehat{\tfrac{\partial G(\boldsymbol{x},\boldsymbol{\mu})}{\partial y_{vi}}} = \tfrac{1}{T}\textstyle\sum_{l=1}^{T}(C([\boldsymbol{x}^l]_{-(v,i)},\boldsymbol{\mu}) - C([\boldsymbol{x}^l]_{+(v,i)},\boldsymbol{\mu})), \quad (20)$$

where $[\boldsymbol{x}^l]_{-(v,i)}$, $[\boldsymbol{x}^l]_{+(v,i)}$ are equal to vector $\boldsymbol{x}$ with the (v,i)-th coordinate set to 0 and 1, respectively, and w.r.t μ_e^r:

$$\widehat{\tfrac{\partial G(\boldsymbol{x},\boldsymbol{\mu})}{\partial \mu_e^r}} = \tfrac{1}{T\mu_e^r}\textstyle\sum_{l=1}^{T} \tfrac{\partial C_e^r(\rho_e^r([\boldsymbol{x}^l],\mu_e^r))}{\partial \rho_e^r([\boldsymbol{x}^l],\mu_e^r)} \cdot \rho_e^r([\boldsymbol{x}^l],\mu_e^r). \quad (21)$$

According to Calinescu et al. [26], for the (with high probability) $1 - 1/e$ approximation ratio, $O((|V||\mathcal{C}|)^2 \ln(|V||\mathcal{C}|))$ samples suffice. There are other ways to estimate the gradient, e.g., via a Taylor expansion [13]. This more efficient, so we also use it in Sec. V. We refer readers to [13] for more details.

E. Swap Rounding

We review swap rounding [37], which is a probabilistic rounding step. Given a fractional $\boldsymbol{y}_K$, it can be written as a convex combination of some integer vectors $\boldsymbol{B}_l$, i.e., $\boldsymbol{y}_K = \sum_{l=1}^{L} \beta_l \boldsymbol{B}_l$, where $\sum_{l=1}^{L} \beta_l = 1$, $\beta_l \geq 0$, and $\boldsymbol{B}_l \in \mathcal{D}$. By construction, each $\boldsymbol{B}_l$ is maximal. This algorithm iteratively merges these $\boldsymbol{B}_l$, each iteration one $\boldsymbol{B}_l$, to produce a new integer solution $\boldsymbol{x}_l$, until $\boldsymbol{x}_l$ equal to $\boldsymbol{B}_{l+1}$. If $\boldsymbol{x}_{l'}$ differs $\boldsymbol{B}_{l'+1}$ by an item i in v, the item i replaces another different item j in $\boldsymbol{B}_{l'+1}$ with probability proportional to $\sum_{l=1}^{l'} \beta_l$, or another different item j in $\boldsymbol{B}_{l'+1}$ replaces item i in $\boldsymbol{x}_{l'}$ with probability proportional to $\beta_{l'+1}$. Swap rounding ensures that the objective does not decrease in expectation during rounding, i.e.,

$$\mathbb{E}[G(\boldsymbol{x}_K, \boldsymbol{\mu}_K)] \geq G(\boldsymbol{y}_K, \boldsymbol{\mu}_K). \quad (22)$$

The algorithm terminates in at most $O(|V||\mathcal{C}|)$ steps.

topologies	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{Q}\rvert$	c_v
ER	100	1042	100	1000	4	2
ER-20Q	100	1042	100	1000	20	2
star	100	198	100	1000	4	2
HC	128	896	100	1000	4	2
HC-20Q	128	896	100	1000	20	2
dtelekom	68	546	100	1000	4	2
abilene	9	26	100	1000	4	2
geant	22	66	100	1000	4	2

TABLE II: Graph Topologies and Experiment Parameters.

F. Time Complexity

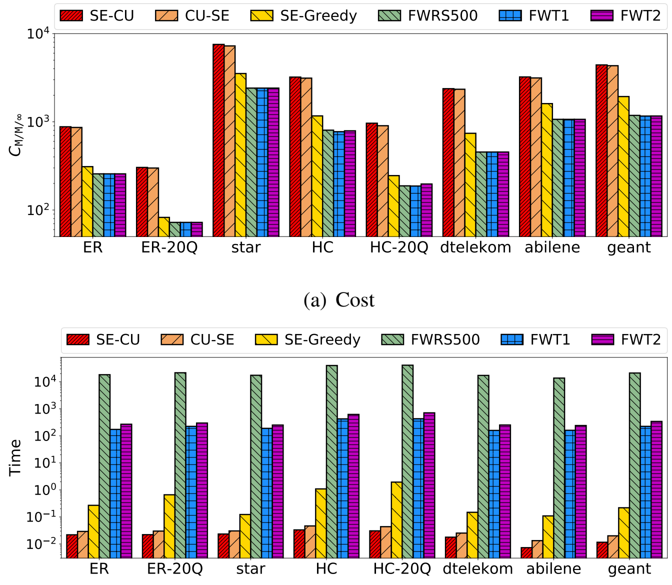
To ensure Thm. 4 holds, the number of samples T used for sample-based estimator of gradient is $O((|V||\mathcal{C}|)^2 \ln(|V||\mathcal{C}|))$ [13]. Each sample requires at most $O(|E||\mathcal{R}|)$ operations. Given a gradient, (15) requires polynomial time in the number of constraints and variables, which are $O(|V||\mathcal{C}| + |E||\mathcal{R}|)$. We iterate (15) at most $O(|V||\mathcal{C}|)$ times [13]. The rounding schemes presented in Sec. IV-E are also poly-time, i.e., at most $O(|V||\mathcal{C}|)$ steps. In summary, the overall time complexity of our algorithm is $O(|E||\mathcal{R}|(|V||\mathcal{C}|)^3 \ln(|V||\mathcal{C}|))$.

V. Experiments

1) Experiment Setting: We execute our algorithms on Erdős-Rényi (`ER`), `star`, hypercube (`HC`), Deutsche Telekom (`dtelekom`), `GEANT`, and `Abilene` backbone networks [38]. The graph parameters of different topologies are shown in Tab. II. Each node $v \in V$ has c_v storage to cache item from a catalog of size $|\mathcal{C}|$. Each item $i \in \mathcal{C}$ is stored permanently in one designated server $\mathcal{S}_i$ which is picked uniformly at random (u.a.r.) from V. Also, we u.a.r. select $|\mathcal{Q}|$ nodes from V as query nodes. Each of them generates around $\lfloor|\mathcal{R}|/|\mathcal{Q}|\rfloor$ requests. For each request type $r \in \mathcal{R}$, rate λ^r is uniformly distributed over $[1.0, 2.0]$. The item i^r requested by r is chosen from catalog $\mathcal{C}$ via a power law distribution with exponent 1.2. The path p^r is the shortest path between the query node $p_1^r \in \mathcal{Q}$ and designated server $p_K^r \in \mathcal{S}_{i^r}$. We set $\mu_e = 200.0$ at each edge e, and $\epsilon = 0.1$. Cost functions $c_e(n_e^r)$ are moments of the queue size $\mathbb{E}[(n_e^r)^k]$, where $k = 1, 2, 3, 4$.

We conduct two types of experiments. (i) In the *offline* setting, we compute the expected costs $C_{\texttt{M/M/}\infty}$ and $C_{\texttt{M/M/1c}}$ according to (8a) and (10a), respectively. (ii) In the online setting: we simulate packets in `M/M/1c` and $\texttt{M/M/}\infty$ queues network, and compute the time-average cost. More specifically, we monitor queues status at epochs t_s of a Poisson process with rate 1.0, leveraging PASTA [39], for 5000 time slots. For N measurements, the time average cost is: $\bar{C}_\cdot = \frac{1}{N}\sum_{s=0}^{N}\sum_{e\in E}\sum_{r\in\mathcal{R}_e} c_e^r(n_e^r(t_s))$, where $\cdot \in \{\texttt{M/M/}\infty, \texttt{M/M/1c}\}$ indicates on the type of queues simulated, and $n_e^r(t_s)$ is queue size at epoch t_s.

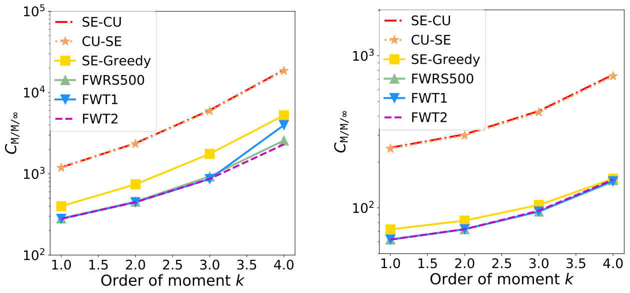
2) Cache and Service Rate Algorithms: We compare to both online and offline algorithms. Offline algorithms are: (a) *Service Equally-Cache Uniformly* (SE-CU): first equally assign service rates for $\mathcal{R}_e$ over all $e \in E$ and then uniformly place items in each node. (b) *Cache Uniformly-Service Equally* (CU-SE): first uniformly place items in each node and then



(a) Cost

(b) Time

Fig. 3: Cost and time for different topologies and algorithms for quadratic costs.



(a) dtelekom

(b) ER20Q

Fig. 4: Cost at different order of moments. Our algorithms achieve the lowest cost in different cost functions.

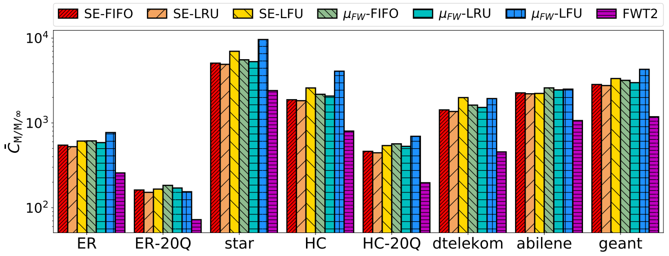


Fig. 5: Time average cost for different typologies and algorithms under quadratic costs.

equally assign service rates for responses passing through edge e for all $e \in E$. Note that service rates depend on item placements in CU-SE. (c) *Service Equally-Greedy* (SE-Greedy): first equally assign service rates for $\mathcal{R}_e$ over all $e \in E$ and then use the classic greedy algorithm [25] for item placements. (d) *Frank-Wolfe with 500 Random Samples* (FWRS500): Alg. 1 with gradient estimated by 500 random samples. (e) *Frank-Wolfe with 1st/2nd order Taylor expansion* (FWT1/FWT2): Alg. 1 with gradient estimated by the first and the second order Taylor expansion respectively (c.f. [13]).

We also consider online algorithms, in which service rates are determined in advance and item placements change dynamically. As in the offline algorithms, we consider two service strategies: (a) *Service Equally* (SE): equally assign service rates for $\mathcal{R}_e$ over all $e \in E$, (b) μ_{FW}: service rates $\boldsymbol{\mu}$ calculated by Alg. 1. We combine these with item placements based on path replication [8]: when responses are back-propagated over the reverse path, nodes they encounter store requested items, evicting items via LRU, LFU, or FIFO eviction policies.

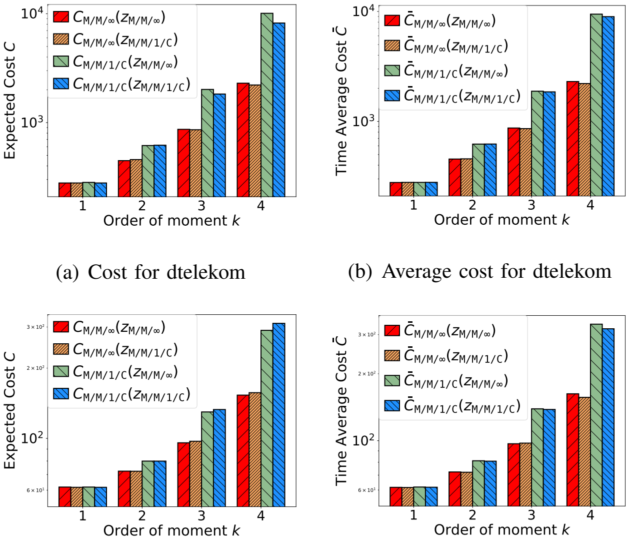
3) Results: **Different Topologies.** The cache cost $C_{\mathtt{M/M/}\infty}$ and the running time generated by different algorithms for quadratic costs is shown in Fig. 3(a) and 3(b). SE-Greedy improves over SE-CU and CU-SE, nevertheless, FW algorithms yield further improvements. As in [13], Taylor approximations are faster than sampling, without a loss in performance.

Different Cost Functions. The impact of the cost function exponent on different algorithms is shown over dtelekom and ER-20Q in Fig. 4(a) and 4(b). Consistently with Fig. 3, FW outperforms competitors, with SE-Greedy being a close second. FWT1 performance degrades at the 4th moment in Fig. 4(a) due to the poor quality of the 1st order Taylor expansion.

Online Algorithms. Fig. 5 compares FWT2 to online algorithms under quadratic costs over $\mathtt{M/M/}\infty$ queues. FWT2 achieves the lowest time average costs $\bar{C}_{\mathtt{M/M/}\infty}$. Eviction algorithms with μ_{FW} are worse than SE most of the time. This means good performance of FWT2 comes from joint optimization. Note that time average costs of FWT2 in Fig. 5 and expected costs in Fig. 3(a) are almost identical, which verifies the reliability of our experiments from another perspective.

Comparing $\mathtt{M/M/1c}$ and $\mathtt{M/M/}\infty$ Queues. Finally, we confirm the quality of our two approximations of $\mathtt{M/M/1c}$ queues experimentally. Our goal is to (i) understand how well expected cost objectives (8a) and (10a) capture the time average costs $\bar{C}_{\mathtt{M/M/1c}}$, and (ii) assess the quality of solutions $\boldsymbol{z}_{\mathtt{M/M/1c}}$ and $\boldsymbol{z}_{\mathtt{M/M/}\infty}$ to Problems (10) and (8), respectively.

To that end, we plot both the expected cost objectives $C_{\mathtt{M/M/1c}}$, $C_{\mathtt{M/M/}\infty}$, as well as the time averages $\bar{C}_{\mathtt{M/M/1c}}$, $\bar{C}_{\mathtt{M/M/}\infty}$ for the two inputs $\boldsymbol{z}_{\mathtt{M/M/1c}}$ and $\boldsymbol{z}_{\mathtt{M/M/}\infty}$ in Fig. 6. We make the following broad observations. First, expected costs (Fig. 6(a) and 6(c)) are almost identical to time average costs (Fig. 6(b) and 6(d)). This is anticipated for $\mathtt{M/M/}\infty$ queues, that form a Jackson network, but is not obvious for $\mathtt{M/M/1c}$ queues. Second, cost functions $C_{\mathtt{M/M/}\infty}$ and $C_{\mathtt{M/M/1c}}$ differ, and this difference becomes more pronounced as k increases; this is again anticipated by Thm. 1, as the stochastic domination becomes looser for larger k. Nevertheless, the *solutions* $\boldsymbol{z}_{\mathtt{M/M/1c}}$ *and* $\boldsymbol{z}_{\mathtt{M/M/}\infty}$ *exhibit almost identical behavior w.r.t all four objectives*. For example, $C_{\mathtt{M/M/1c}}(\boldsymbol{z}_{\mathtt{M/M/}\infty}) \approx C_{\mathtt{M/M/1c}}(\boldsymbol{z}_{\mathtt{M/M/1c}}) \approx \bar{C}_{\mathtt{M/M/1c}}(\boldsymbol{z}_{\mathtt{M/M/1c}}) \approx \bar{C}_{\mathtt{M/M/1c}}(\boldsymbol{z}_{\mathtt{M/M/}\infty})$. This means that, even



(a) Cost for dtelekom

(b) Average cost for dtelekom

(c) Cost for ER-20Q

(d) Average cost for ER-20Q

Fig. 6: Expected costs and time-average costs in $\mathtt{M/M/\infty}$ v.s. $\mathtt{M/M/1c}$ queue networks. The two approximate models obtain similar, and good, solutions.

	ER	ER-20Q	star	HC	HC-20Q	DT	ab/ne	geant
$C_{\mathtt{M/M/1/C}}(\boldsymbol{z}_{\mathtt{M/M/\infty}})$	332	79.9	3827	1112	227	616	1540	1697
$\bar{C}_{\mathtt{M/M/1/C}}(\boldsymbol{z}_{\mathtt{M/M/\infty}})$	334	81.2	3880	1202	244	622	1714	1860
$C_{\mathtt{M/M/1/C}}(\boldsymbol{z}_{\mathtt{M/M/1/C}})$	332	79.8	3842	1115	212	620	1570	1707
$\bar{C}_{\mathtt{M/M/1/C}}(\boldsymbol{z}_{\mathtt{M/M/1/C}})$	341	80.9	3871	1199	228	623	1759	1884

TABLE III: Expected costs and time-average costs under different topologies for quadratic cost functions.

though the two objectives are not the same, the quality of the solutions that they produce is quite similar. We also observe this in Table III, where these numbers are shown for quadratic objectives across topologies.

VI. Conclusion

We model a cache network as a system of counting queues in which identical packets merge when collocated. We propose an offline algorithm; even though item placements and service rate assignments should not be presumed as static, as they depend on the demand as solutions to Problem (11), fully adaptive algorithms would be interesting to study. Merging on the basis of items, rather than request types, or even stopping requests early when they encounter queues that store an item, rather than caches, are interesting practical directions. Both are harder to analyze, but intuition gained in our setting may be applicable in these contexts too.

Appendix

A. Jackson Network

Let the state of a network of $\mathtt{M/M/\infty}$ queues be $\mathbf{n} = [n_e^r]_{e\in E, r\in\mathcal{R}_e}$, where n_e^r is the number of responses of type r on edge e. Then, for each $r \in \mathcal{R}$, the corresponding network is a Jackson network [14], [35] and, in particular:

Lemma 4. *The joint distribution in steady state has following product form:* $\pi(\boldsymbol{n}) = \prod_{e\in E}\prod_{r\in\mathcal{R}}\pi_e^r(n_e^r)$, *where* $\pi_e^r(n_e^r) = \frac{(\rho_e^r)^{n_e^r}}{(n_e^r)!}e^{-\rho_e^r}$, $n_e^r \in \mathbb{N}$.

B. Proof of Theorem 1

We first state two auxiliary lemmas.

Lemma 5. $m^k_{\mathtt{M/M/1c}}(\rho)$ *and* $m^k_{\mathtt{M/M/\infty}}(\rho)$ *can be obtained from recurrence relations:*

$$m^{k+1}_{\mathtt{M/M/1c}}(\rho) = \rho m^k_{\mathtt{M/M/1c}}(\rho) + \rho(\rho+1)\frac{\mathrm{d}m^k_{\mathtt{M/M/1c}}(\rho)}{\mathrm{d}\rho},$$
$$m^{k+1}_{\mathtt{M/M/\infty}}(\rho) = \rho m^k_{\mathtt{M/M/\infty}}(\rho) + \rho\frac{\mathrm{d}m^k_{\mathtt{M/M/\infty}}(\rho)}{\mathrm{d}\rho}.$$

Proof. The $m^k_{\mathtt{M/M/\infty}}$ statement is due to Riordan (see Sec. 3, pp.105-106 in [40]). The $m^k_{\mathtt{M/M/1c}}$ case follows similarly. □

Lemma 6. *Both* $m^k_{\mathtt{M/M/1c}}(\rho)$ *and* $m^k_{\mathtt{M/M/\infty}}(\rho)$ *are polynomials, i.e., :* $m^k_{\mathtt{M/M/1c}}(\rho) = \sum_{i=1}^k \beta_i^k\rho^i$, $m^k_{\mathtt{M/M/\infty}}(\rho) = \sum_{i=1}^k \alpha_i^k\rho^i$, *where* $\beta_i^k, \alpha_i^k > 0$, *and* $\frac{\beta_i^k}{\alpha_i^k} = i!$

Proof. We prove this by induction. For $k = 1$, this follows from (3) (also Lem. 5 for $k = 0$). Suppose it holds for $k = \ell$, i.e., $m^\ell_{\mathtt{M/M/1c}}(\rho) = \sum_{i=1}^\ell i!\alpha_i^\ell\rho^i$, and $m^\ell_{\mathtt{M/M/\infty}}(\rho) = \sum_{i=1}^\ell \alpha_i^\ell\rho^i$. Then, when $k = \ell+1$, by Lemma 5:

$$m^{\ell+1}_{\mathtt{M/M/1c}}(\rho) = \alpha_1^\ell\rho + \sum_{i=2}^\ell i!(\alpha_{i-1}^\ell + i\alpha_i^\ell)\rho^i + (\ell+1)!\alpha_\ell^\ell\rho^{\ell+1}$$
$$m^{\ell+1}_{\mathtt{M/M/\infty}}(\rho) = \alpha_1^\ell\rho + \sum_{i=2}^\ell(\alpha_{i-1}^\ell + i\alpha_i^\ell)\rho^i + \alpha_\ell^\ell\rho^{\ell+1}$$

Comparing terms, we have $\frac{\beta_i^{\ell+1}}{\alpha_i^{\ell+1}} = i!$. □

To prove Thm. 1, observe that by Lemma 6, $\frac{m^k_{\mathtt{M/M/\infty}}(\rho)}{m^k_{\mathtt{M/M/1c}}(\rho)} = \frac{\sum_{i=1}^k \alpha_i^k\rho^i}{\sum_{i=1}^k i!\alpha_i^k\rho^i}$. Hence, $\frac{\sum_{i=1}^k \alpha_i^k\rho^i}{\sum_{i=1}^k k!\alpha_i^k\rho^i} \le \frac{m^k_{\mathtt{M/M/\infty}}(\rho)}{m^k_{\mathtt{M/M/1c}}(\rho)} \le \frac{\sum_{i=1}^k \alpha_i^k\rho^i}{\sum_{i=1}^k \alpha_i^k\rho^i}$, which implies $\frac{1}{k!} \le \frac{m^k_{\mathtt{M/M/\infty}}(\rho)}{m^k_{\mathtt{M/M/1c}}(\rho)} \le 1$. □

C. Proof of Theorem 2 [Sketch]

Observe that $\textsc{MinCost}_{\mathtt{M/M/\infty}}$ and $\textsc{MinCost}_{\mathtt{M/M/1c}}$ are identical when $c_e^r(n_e^r) = n_e^r$, i.e., when the cost is the queue size: by (3), both expected costs are equal to ρ_e^r. We reduce the (NP-hard) fixed routing cost problem by Ioannidis and Yeh [10] to this problem. To do so, if an edge has cost w_{uv}, we set $\mu_{uv} = |\mathcal{R}_{uv}|/w_{uv}$ and $\epsilon = 1/w_{uv}$. Then the service rate μ_{uv}^r at each queue on edge (u,v) is exactly $1/w_{uv}$ (i.e., $\mathcal{D}_\mu$ is a singleton), and both $\textsc{MinCost}_{\mathtt{M/M/\infty}}$ and $\textsc{MinCost}_{\mathtt{M/M/1c}}$ coincide with the fixed cost routing problem. □

D. Proof of Lemma 3

By Eq. (9), the expected costs of $\mathtt{M/M/1c}$ queues are:

$$\mathbb{E}_{\mathtt{M/M/1c}}[c_e^r(n_e^r)] = c_e^r(0) + \sum_{n=0}^{\infty}(c_e^r(n+1) - c_e^r(n))(\frac{\rho_e^r}{\rho_e^r+1})^{n+1}.$$

Hence, $\frac{\mathrm{d}\mathbb{E}_{\mathtt{M/M/1c}}[c_e^r(n_e^r)]}{\mathrm{d}\rho_e^r} = \sum_{n=0}^\infty(c_e^r(n+1) - c_e^r(n)) \cdot \frac{(n+1)(\rho_e^r)^n}{(\rho_e^r+1)^{n+2}} \ge 0$. Moreover, $\frac{\mathrm{d}^2\mathbb{E}_{\mathtt{M/M/1c}}[c_e^r(n_e^r)]}{\mathrm{d}(\rho_e^r)^2} = \sum_{n=0}^\infty \Delta_n$, where $\Delta_n = (c_e^r(n+1) - c_e^r(n))(n+1)\frac{n(\rho_e^r)^{n-1} - 2(\rho_e^r)^n}{(\rho_e^r+1)^{n+3}}$.

By Assumption 1, for $n_0 \equiv \lfloor 2\rho_e^r \rfloor$ we have that $\Delta_n \geq (c_e^r(n_0+1) - c_e^r(n_0))(n+1)\frac{n(\rho_e^r)^{n-1} - 2(\rho_e^r)^n}{(\rho_e^r+1)^{n+3}}$, for all $n \in \mathbb{N}$. Hence, $\frac{\mathrm{d}^2 \mathbb{E}_{\mathtt{M/M/1c}}[c_e^r(n_e^r)]}{\mathrm{d}(\rho_e^r)^2} \geq (c_e^r(n_0+1) - c_e^r(n_0)) \sum_{n=0}^{\infty} \big[(n+1)n\frac{(\rho_e^r)^{n-1}}{(\rho_e^r+1)^{n+3}} - 2(n+1)\frac{(\rho_e^r)^n}{(\rho_e^r+1)^{n+3}}\big] = 0$. Thus, $\mathbb{E}_{\mathtt{M/M/1c}}[c_e^r(n_e^r)]$ is non-decreasing and convex w.r.t. ρ_e^r. Similarly, by Eq. (6):

$$\mathbb{E}_{\mathtt{M/M/}\infty}[c_e^r(n_e^r)] = c_e^r(0) + \sum_{n=0}^{\infty}(c_e^r(n+1) - c_e^r(n))e^{-\rho_e^r}\sum_{l=n+1}^{\infty}\frac{(\rho_e^r)^l}{l!}.$$

Hence, $\frac{\mathrm{d}\mathbb{E}_{\mathtt{M/M/}\infty}[c_e^r(n_e^r)]}{\mathrm{d}\rho_e^r} = \sum_{n=0}^{\infty}(c_e^r(n+1) - c_e^r(n)) \cdot e^{-\rho_e^r}\frac{(\rho_e^r)^n}{n!} \geq 0$. Moreover, $\frac{\mathrm{d}^2\mathbb{E}_{\mathtt{M/M/}\infty}[c_e^r(n_e^r)]}{\mathrm{d}(\rho_e^r)^2} = \sum_{n=0}^{\infty}\Delta_n$, where $\Delta_n = (c_e^r(n+1) - c_e^r(n))(-e^{-\rho_e^r}\frac{(\rho_e^r)^n}{n!} + e^{-\rho_e^r}\frac{n(\rho_e^r)^{n-1}}{n!})$. By Assumption 1, for $n_0 \equiv \lfloor \rho_e^r \rfloor$ we have that $\Delta_n \geq (c_e^r(n_0+1) - c_e^r(n_0))(-e^{-\rho_e^r}\frac{(\rho_e^r)^n}{n!} + e^{-\rho_e^r}\frac{n(\rho_e^r)^{n-1}}{n!})$, for all $n \in \mathbb{N}$. Hence, $\frac{d^2\mathbb{E}_{\mathtt{M/M/}\infty}[c_e^r(n_e^r)]}{d(\rho_e^r)^2} \geq (c_e^r(n_0+1) - c_e^r(n_0)) \cdot (\sum_{n=0}^{\infty} -e^{-\rho_e^r}\frac{(\rho_e^r)^n}{n!} + \sum_{n=1}^{\infty} e^{-\rho_e^r}\frac{(\rho_e^r)^{n-1}}{(n-1)!}) = 0$. Thus, $\mathbb{E}_{\mathtt{M/M/}\infty}[c_e^r(n_e^r)]$ is non-decreasing and convex w.r.t. ρ_e^r. □

E. Proof of Theorem 3

Let function $F(S, \boldsymbol{\mu}) \triangleq F(\boldsymbol{x}_S, \boldsymbol{\mu})$, $S = \{\mathrm{supp}(\boldsymbol{x})\}$. We first introduce an auxiliary lemma:

Lemma 7. *If the expected cost functions C_e^r are non-decreasing and convex, the set function $F(S, \boldsymbol{\mu})$ is: (a) non-decreasing concave on $\boldsymbol{\mu}$ and (b) non-decreasing submodular on set S.*

Proof. For convenience, we replace subscripts e and superscripts r by subscript $i \in E \times \sum_e \mathcal{R}_e$. By Lemma 3, $C_i(\rho_i)$ is non-decreasing convex w.r.t. ρ_i. And, $\rho_i = \frac{\lambda_i}{\mu_i}$ is decreasing convex w.r.t. μ_i. Hence, by Eq. (3.10), p.84 of [41], we have that $\frac{\partial C_i(\boldsymbol{x}, \mu_i)}{\partial \mu_i} \leq 0$, $\frac{\partial^2 C_i(\boldsymbol{x}, \mu_i)}{\partial \mu_i^2} \geq 0$. Hence, the first derivative of $F(S, \boldsymbol{\mu})$ w.r.t. μ_i is: $\frac{\partial F(S, \boldsymbol{\mu})}{\partial \mu_i} = -\frac{\partial C(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i} \geq 0$, and the second derivative w.r.t. μ_i, μ_j is:

$$\frac{\partial^2 F(S, \boldsymbol{\mu})}{\partial \mu_i \partial \mu_j} = -\frac{\partial^2 C(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i \partial \mu_j} = \begin{cases} 0 & i \neq j \\ -\frac{\partial^2 C_i(\boldsymbol{x}, \mu_i)}{\partial \mu_i^2} & i = j \end{cases} \leq 0,$$

so $\nabla_{\boldsymbol{\mu}} F(S, \boldsymbol{\mu}) \geq \mathbf{0}$ and $\nabla^2_{\boldsymbol{\mu}} F(S, \boldsymbol{\mu}) \preceq 0$. And $F(S, \boldsymbol{\mu})$ is non-decreasing and concave on $\boldsymbol{\mu}$. We know that C_i is non-decreasing and convex w.r.t. ρ_i. The first derivative of $F(S, \boldsymbol{\mu})$ w.r.t. ρ_i is: $\frac{\partial F(S, \boldsymbol{\mu})}{\partial \rho_i} = -\frac{\partial C(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \rho_i} \leq 0$, and the second derivative w.r.t. ρ_i, ρ_j is:

$$\frac{\partial^2 F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \rho_i \partial \rho_j} = -\frac{\partial^2 C(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \rho_i \partial \rho_j} = \begin{cases} 0 & i \neq j \\ -\frac{\partial^2 C_i(\boldsymbol{x}, \mu_i)}{\partial \rho_i^2} & i = j \end{cases} \leq 0,$$

so $\nabla_{\boldsymbol{\rho}} F(\boldsymbol{x}, \boldsymbol{\mu}) \leq \mathbf{0}$, $\nabla^2_{\boldsymbol{\rho}} F(\boldsymbol{x}, \boldsymbol{\mu}) \preceq 0$. Furthermore, $F(S, \boldsymbol{\mu})$ is non-increasing and concave on $\boldsymbol{\rho}$. By Corollary 1 in [13], $F(S, \boldsymbol{\mu})$ is non-decreasing and submodular on set S. □

By Lemma 7:

$$\frac{\partial G(\boldsymbol{y}, \boldsymbol{\mu})}{\partial \mu_i} = \sum_{\boldsymbol{x} \in \{0,1\}^{|V||\mathcal{C}|}} \frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i} \times \prod_{(v,i) \in V \times \mathcal{C}} y_{vi}^{x_{vi}}(1 - y_{vi})^{1 - x_{vi}} \geq 0,$$

and

$$\frac{\partial G(\boldsymbol{y}, \boldsymbol{\mu})}{\partial y_i} = \mathbb{E}_y[F(\boldsymbol{x}, \boldsymbol{\mu})|x_i = 1] - \mathbb{E}_y[F(\boldsymbol{x}, \boldsymbol{\mu})|x_i = 0] \geq 0.$$

Thus G is non-decreasing in both $\boldsymbol{\mu}$ and $\boldsymbol{y}$. By Lemma 7, we get:

$$\frac{\partial^2 G(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i \partial \mu_j} = \sum_{\boldsymbol{x} \in \{0,1\}^{|V||\mathcal{C}|}} \frac{\partial^2 F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i \partial \mu_j} \times \prod_{(v,i) \in V \times \mathcal{C}} y_{vi}^{x_{vi}}(1 - y_{vi})^{1 - x_{vi}} \leq 0,$$

while, as shown in [26],

$$\begin{aligned}\frac{\partial^2 G(\boldsymbol{y}, \boldsymbol{\mu})}{\partial y_i \partial y_j} = & \mathbb{E}_y[F(\boldsymbol{x}, \boldsymbol{\mu})|x_i{=}1, x_j{=}1] - \mathbb{E}_y[F(\boldsymbol{x}, \boldsymbol{\mu})|x_i{=}1, x_j{=}0] \\ & - \mathbb{E}_y[F(\boldsymbol{x}, \boldsymbol{\mu})|x_i{=}0, x_j{=}1] + \mathbb{E}_y[F(\boldsymbol{x}, \boldsymbol{\mu})|x_i{=}0, x_j{=}0] \\ & \leq 0. \end{aligned} \tag{23}$$

Then, $\frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i} = -\frac{\partial C_i(\rho_i)}{\partial \rho_i}\frac{\partial \rho_i}{\partial \mu_i} = \frac{\partial C_i(\rho_i)}{\partial \rho_i}\frac{\rho_i}{\mu_i} \geq 0$, and $\frac{\partial}{\partial \rho_i}\frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i} = \frac{\partial^2 C_i(\rho_i)}{\partial \rho_i^2}\frac{\rho_i}{\mu_i} + \frac{\partial C_i(\rho_i)}{\partial \rho_i}\frac{1}{\mu_i} \geq 0$. So, $\frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i}$ is non-decreasing w.r.t. ρ_i. Since ρ_i is non-increasing w.r.t. $\boldsymbol{x}$, $\frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i}$ is non-increasing w.r.t. $\boldsymbol{x}$. Then $\frac{\partial^2 G(\boldsymbol{y}, \boldsymbol{\mu})}{\partial \mu_i \partial y_j} = \mathbb{E}_y[\frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i}|x_j = 1] - \mathbb{E}_y[\frac{\partial F(\boldsymbol{x}, \boldsymbol{\mu})}{\partial \mu_i}|x_j = 0] \leq 0$. Hence, all of the entries of $G(\boldsymbol{x}, \boldsymbol{\mu})$'s Hessian w.r.t. both $\boldsymbol{\mu}$ and $\boldsymbol{x}$ are non-positive, and G is DR-submodular [15]. □

F. Proof of Theorem 4

We begin by stating a lemma on the quality of the output of Alg. 1, assuming that the estimate $\widehat{\nabla G}$ is produced via the sampling method outlined in Sec. IV-D.

Lemma 8. *For a fixed number of iterations K which is large enough, and constant stepsize $\gamma_k = \gamma = K^{-1}$, Alg. 1 provides the following approximation guarantee with high probability:*

$$G(\boldsymbol{z}_K) \geq (1 - \tfrac{1}{e})G(\boldsymbol{z}^*), \tag{24}$$

where $\boldsymbol{z}_K$ is output of Alg. 1, $\boldsymbol{z}^*$ is an optimal solution to Problem (16)*.*

Proof. G is DR-submodular shown in Lemma 3, domain $\tilde{\mathcal{D}} \times \mathcal{D}_\mu$ is a down-closed convex set, and Lipschitz parameter of ∇G is $2C(\mathbf{0}, \boldsymbol{\epsilon})$ because $||\nabla^2 G(\boldsymbol{z})||$ is bounded by $2C(\mathbf{0}, \boldsymbol{\epsilon})$ according to (23). With above conditions, Corollary 1 by Bian et al. [29] states: $G(\boldsymbol{z}_K) \geq (1 - \frac{1}{e})G(\boldsymbol{z}^*) - \frac{L}{2K}$, where $L = 2C(\mathbf{0}, \boldsymbol{\epsilon})$ is a finite constant. Calinescu et al. [26] show that for large enough K, the offset $\frac{L}{2K}$ can be omitted and (24) still holds with high probability. The term "with high probability", due to sample-based estimation of ∇G, and means probability at least $1 - 1/|V||C|$. □

To conclude the proof of Theorem 4, we have: $\mathbb{E}[F(\boldsymbol{x}_K, \boldsymbol{\mu}_K)] \overset{\text{Eq. (14)}}{=} \mathbb{E}[G(\boldsymbol{x}_K, \boldsymbol{\mu}_K)] \overset{\text{Eq. (22)}}{\geq} G(\boldsymbol{y}_K, \boldsymbol{\mu}_K) \overset{\text{Lem. 8}}{\geq} (1 - \frac{1}{e})G(\boldsymbol{y}^*, \boldsymbol{\mu}^*) \geq (1 - \frac{1}{e})F(\boldsymbol{x}^*, \boldsymbol{\mu}^*)$, where $\boldsymbol{x}^*$ and $\boldsymbol{\mu}^*$ is an optimal solution to (12), $\boldsymbol{y}^*$ is an optimal solution to (16), $\boldsymbol{y}_K$ and $\boldsymbol{\mu}_K$ is the output of Frank-Wolfe variant algorithm, and $\boldsymbol{x}_K$ is the integer solution after rounding. The first equation holds because F and G are equal under integer arguments $\boldsymbol{x}_K$. The last inequality holds because (16) has a larger feasible region. □

References

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009, pp. 1–12.

[2] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, “A data-oriented (and beyond) network architecture,” *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 181–192, 2007.

[3] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, “Network of information (netinf)–an information-centric networking architecture,” *Computer Communications*, vol. 36, no. 7, pp. 721–735, 2013.

[4] D. A. Farber, R. E. Greer, A. D. Swart, and J. A. Balter, “Internet content delivery network,” Nov. 25 2003, uS Patent 6,654,807.

[5] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang, “An untold story of middleboxes in cellular networks,” in *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4. ACM, 2011, pp. 374–385.

[6] A. Anand, V. Sekar, and A. Akella, “Smartre: an architecture for coordinated network-wide redundancy elimination,” in *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4. ACM, 2009, pp. 87–98.

[7] X. Zhang, J. Liu, B. Li, and Y.-S. Yum, “Coolstreaming/donet: A data-driven overlay network for peer-to-peer live media streaming,” in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 3. IEEE, 2005, pp. 2102–2111.

[8] E. Cohen and S. Shenker, “Replication strategies in unstructured peer-to-peer networks,” in *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4. ACM, 2002, pp. 177–190.

[9] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, “Femtocaching: Wireless content delivery through distributed caching helpers,” *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[10] S. Ioannidis and E. Yeh, “Adaptive caching networks with optimality guarantees,” in *ACM SIGMETRICS Performance Evaluation Review*, vol. 44, no. 1. ACM, 2016, pp. 113–124.

[11] ——, “Jointly optimal routing and caching for arbitrary network topologies,” *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1258–1275, 2018.

[12] S. E. Hajri and M. Assaad, “Energy efficiency in cache-enabled small cell networks with adaptive user clustering,” *IEEE Transactions on Wireless Communications*, vol. 17, no. 2, pp. 955–968, 2017.

[13] M. Mahdian, A. Moharrer, S. Ioannidis, and E. Yeh, “Kelly cache networks,” in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 217–225.

[14] F. P. Kelly, *Reversibility and stochastic networks*. Cambridge University Press, 2011.

[15] A. Bian, K. Levy, A. Krause, and J. M. Buhmann, “Continuous dr-submodular maximization: Structure and algorithms,” in *Advances in Neural Information Processing Systems*, 2017, pp. 486–496.

[16] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, “Optimal content placement for a large-scale vod system,” in *Proceedings of the 6th International COnference*. ACM, 2010, p. 4.

[17] I. Baev, R. Rajaraman, and C. Swamy, “Approximation algorithms for data placement problems,” *SIAM Journal on Computing*, vol. 38, no. 4, pp. 1411–1429, 2008.

[18] Y. Bartal, A. Fiat, and Y. Rabani, “Competitive algorithms for distributed data management,” *Journal of Computer and System Sciences*, vol. 51, no. 3, pp. 341–358, 1995.

[19] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko, “Tight approximation algorithms for maximum general assignment problems,” in *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial and Applied Mathematics, 2006, pp. 611–620.

[20] S. Borst, V. Gupta, and A. Walid, “Distributed caching algorithms for content distribution networks,” in *2010 Proceedings IEEE INFOCOM*. Citeseer, 2010, pp. 1–9.

[21] 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 *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 936–944.

[22] S. Shukla and A. A. Abouzeid, “Proactive retention aware caching,” in *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE, 2017, pp. 1–9.

[23] J. Li, T. K. Phan, W. K. Chai, D. Tuncer, G. Pavlou, D. Griffin, and M. Rio, “Dr-cache: Distributed resilient caching with latency guarantees,” in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 441–449.

[24] F. Zafari, J. Li, K. K. Leung, D. Towsley, and A. Swami, “Optimal energy tradeoff among communication, computation and caching with qoi-guarantee,” in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–7.

[25] A. Krause and D. Golovin, “Submodular function maximization.” 2014.

[26] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, “Maximizing a monotone submodular function subject to a matroid constraint,” *SIAM Journal on Computing*, vol. 40, no. 6, pp. 1740–1766, 2011.

[27] D. P. Bertsekas, *Nonlinear programming*. Athena scientific Belmont, 1999.

[28] M. Sviridenko, J. Vondrák, and J. Ward, “Optimal approximation for submodular and supermodular optimization with bounded curvature,” *Mathematics of Operations Research*, vol. 42, no. 4, pp. 1197–1218, 2017.

[29] A. A. Bian, B. Mirzasoleiman, J. M. Buhmann, and A. Krause, “Guaranteed non-convex optimization: Submodular maximization over continuous domains,” *arXiv preprint arXiv:1606.05615*, 2016.

[30] A. A. Ageev and M. I. Sviridenko, “Pipage rounding: A new method of constructing algorithms with proven performance guarantee,” *Journal of Combinatorial Optimization*, vol. 8, no. 3, pp. 307–328, 2004.

[31] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-Hall International New Jersey, 1992, vol. 2.

[32] R. G. Gallager, *Stochastic processes: theory for applications*. Cambridge University Press, 2013.

[33] P. G. Harrison and N. M. Patel, *Performance modelling of communication networks and computer architectures (International Computer S*. Addison-Wesley Longman Publishing Co., Inc., 1992.

[34] M. H. Stone, “Applications of the theory of boolean rings to general topology,” *Transactions of the American Mathematical Society*, vol. 41, no. 3, pp. 375–481, 1937.

[35] J. R. Jackson, “Networks of waiting lines,” *Operations research*, vol. 5, no. 4, pp. 518–521, 1957.

[36] M. R. Hestenes, “Optimization theory: the finite dimensional case,” *New York*, 1975.

[37] C. Chekuri, J. Vondrak, and R. Zenklusen, “Dependent randomized rounding via exchange properties of combinatorial structures,” in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 2010, pp. 575–584.

[38] D. Rossi and G. Rossini, “Caching performance of content centric networks under multi-path routing (and more),” *Relatório técnico, Telecom ParisTech*, pp. 1–6, 2011.

[39] R. W. Wolff, “Poisson arrivals see time averages,” *Operations Research*, vol. 30, no. 2, pp. 223–231, 1982.

[40] J. Riordan, “Moment recurrence relations for binomial, poisson and hypergeometric frequency distributions,” *The Annals of Mathematical Statistics*, vol. 8, no. 2, pp. 103–111, 1937.

[41] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.