On the Regret of Online Edge Service Hosting

R Sri Prakash
IIT Bombay
sriprakash@ee.iitb.ac.in

Nikhil Karamchandani
IIT Bombay
nikhilk@ee.iitb.ac.in

Sharayu Moharir
IIT Bombay
sharayum@ee.iitb.ac.in

***Abstract*—We consider the problem of service hosting where a service provider can dynamically rent edge resources via short term contracts to ensure better quality of service to its customers. The service can also be partially hosted at the edge, in which case, customers' requests can be partially served at the edge. The total cost incurred by the system is modeled as a combination of the rent cost, the service cost incurred due to latency in serving customers, and the fetch cost incurred as a result of the bandwidth used to fetch the code/databases of the service from the cloud servers to host the service at the edge. In this paper, we compare multiple hosting policies with regret as a metric, defined as the difference in the cost incurred by the policy and the optimal policy over some time horizon T. In particular we consider the Retro Renting (RR) and Follow The Perturbed Leader (FTPL) policies proposed in the literature and provide performance guarantees on the regret of these policies. We show that under i.i.d stochastic arrivals, RR policy has linear regret while FTPL policy has constant regret. Next, we propose a variant of FTPL, namely Wait then FTPL (W-FTPL), which also has constant regret while demonstrating much better dependence on the fetch cost. We also show that under adversarial arrivals, RR policy has linear regret while both FTPL and W-FTPL have regret $\mathrm{O}(\sqrt{T})$ which is order-optimal.**

I. Introduction

Software as a Service (SaaS) instances like online navigation platforms, Video-on-Demand services, etc., have stringent latency constraints in order to provide a good quality of experience to their customers. While most SaaSs use cloud resources, low latency necessitates the use of storage/computational resources at the edge, i.e., close to the end-user. A service is said to be hosted at the edge if the code and databases needed to serve user queries are stored on the edge servers and requests can be served at the edge. In this work, the service can also be partially hosted at the edge, in which case, customers' requests can be partially served at the edge [1], [2]. [1]

We consider the setting where third-party resources can be rented via short-term contracts to host the service, and the edge hosting status of the SaaS can be dynamically changed over time. If the service is not hosted at the edge, it can be fetched from the cloud servers by incurring a fetch cost. The performance of a hosting policy is a function of the rent cost, the fetch cost, and the quality of experience of the users. We refer to the algorithmic challenge of determining what fraction of the service to host at the edge over time as the *service hosting problem*.

Novel online service hosting policies with provable performance guarantees have been proposed in [3], [4]. The metric of interest in [3], [4] is the *competitive ratio*, defined as the ratio of the cost incurred by an online policy to the cost incurred by an offline optimal policy for the same request arrival sequence. Since the competitive ratio is multiplicative by definition, the absolute value of the difference in the cost incurred by an online policy and the optimal policy can be large even though the competitive ratio of the online policy is close to one. This motivates studying the performance of candidate policies in terms of *regret*, defined as the difference in the cost incurred by the policy and the optimal policy. Regret is a widely used metric in online learning [5], including recently for the caching problem [6], [7] which is closely related to the service hosting problem. In this work, one of our goals is to design hosting policies with provable guarantees in terms of the regret. Another key dimension in the performance guarantees of online policies is the assumption made on the request arrival process. Commonly studied settings include the *stochastic setting* and the *adversarial setting*, and we provide performance guarantees for both settings.

For the service hosting problem, [3] proposed the Retro-Renting (RR) policy and showed that it has a constant competitive ratio with respect to the offline optimal policy. On the other hand, for the closely related caching problem, several policies inspired by the recent advances in online convex optimization have been proposed including Online Gradient Ascent [6], Online Mirror Descent [8], and Follow the Perturbed Leader (FTPL) [9], [7]. In particular, FTPL has been shown to have order-optimal regret for the caching problem in the adversarial setting [9], [7]. In this work, we study regret for the RR and FTPL policies for the service hosting problem, under both the stochastic and adversarial settings. Since RR is a deterministic policy, its regret performance in the adversarial setting is poor. On the other hand, a limitation of FTPL is that it makes hosting decisions agnostic of the fetch cost. As a result, in some cases, FTPL is prone to fetching and evicting the service multiple times in the initial time-slots when its estimate of the request arrival rate is noisy, thus leading to poor performance.

A. Our Contributions

We propose a variant of the FTPL policy called Wait then Follow the Perturbed Leader (W-FTPL). W-FTPL is a randomized policy that takes into account the fetch cost in its decision-making. More specifically, W-FTPL does not fetch the service for an initial wait period which depends on the request arrivals and is an increasing function of the fetch cost. Following the wait period, W-FTPL mimics the FTPL policy.

For i.i.d. stochastic request arrivals and the regret metric, we show that RR is sub-optimal while FTPL and W-FTPL are both order-optimal with respect to the time horizon.

[1]This work is supported by a SERB grant on Leveraging Edge Resources for Service Hosting and a SERB grant on Online Learning with Constraints.

While the regret of FTPL can increase linearly with the fetch cost, the regret of W-FTPL increases at most poly logarithmically. The improved performance of W-FTPL over FTPL is a consequence of the fact that W-FTPL avoids most of the fetches made by FTPL in the initial time-slots and by the end of the wait period, its estimate of the arrival rate is accurate enough to avoid multiple fetches.

For the adversarial setting, we first characterize a fundamental lower bound on the regret of any online hosting policy. In terms of regret, we then show that RR is strictly suboptimal, while FTPL and W-FTPL have order-optimal performance with respect to time.

II. System Setup

We consider a system consisting of a back-end server and an edge-server to serve customers of a service. The back-end server always hosts the service and can serve any requests that are routed to it. In this work, we allow the service to be partially hosted at the edge-server. When the service is partially hosted at the edge, requests are partially served at the edge and partially at the back-end server. Further, the fraction of service hosted at the edge can be changed over time. If the service is not hosted or partially hosted at the edge, parts of the service can be fetched from the back-end server to host at the edge. We consider a time-slotted system.

Sequence of events in each time-slot: In each time-slot, we first make the service hosting decision for that time-slot. Following this, requests may arrive and are served at the edge and/or the back-end server.

Request arrivals: We consider two types of arrival processes. The first where arrivals are stochastic across time-slots with mean μ and the second where the arrival process is generated by an oblivious adversary[2]. In either case, there are up to κ arrivals per slot.

Costs: We model three types of costs.

- *Rent cost* ($\mathcal{C}^{\mathcal{P}}_{R,t}$): The system incurs a cost of c units per time-slot to host the entire service at the edge. For hosting f fraction of service, the system incurs a cost of cf units per time-slot.
- *Service cost* ($\mathcal{C}^{\mathcal{P}}_{S,t}$): This is the cost incurred to use the back-end server to serve (parts of) requests. If f fraction of the service is hosted at the edge, the system incurs a service cost of $g(f)$ units per request, where $g(f)$ is a decreasing function of f. We fix $g(0) = 1$, i.e., service cost is one unit when the service is not hosted at the edge. In [1], [2], it is shown that benefits of partial hosting as limited to the setting where $f+g(f) \leq 1$. Motivated by this, we consider the case where $f+g(f) \leq 1$ for all candidate values of f.
- *Fetch cost* ($\mathcal{C}^{\mathcal{P}}_{F,t}$): The system incurs a cost of $M > 1$ units for each fetch of the entire service from the back-end server to host on the edge-server. For fetching f fraction of service, the system incurs a cost of Mf units.

We consider bounded requests, specifically, let $0 \leq r_t \leq \kappa$ denote the number of request arrivals in time-slot t and $R_t = \sum_{i=1}^{t} r_i$ denote the cumulative requests observed by the end of time-slot t. Further, the rent cost incurred to host the entire service at the edge (c) is less than κ. This is motivated by the fact that if $\kappa < c$, it is strictly sub-optimal to host the entire service at the edge irrespective of the number of arrivals in a time-slot.

Let $\rho^{\mathcal{P}}_t$ denote the edge hosting status of the service in time slot t under policy $\mathcal{P}$. We consider the setting where $\rho^{\mathcal{P}}_t \in \{\alpha_1, \alpha_2, \cdots, \alpha_{K-1}, \alpha_K\}$ and $\rho^{\mathcal{P}}_t = \alpha_i$ means that we host α_i fraction of service at the edge-server in time-slot t. Note that $\alpha_i \in [0,1]$ and in particular $\alpha_1 = 0$, $\alpha_K = 1$ and we also consider $\alpha_i < \alpha_j$ for $i < j$. It follows that $\rho^{\mathcal{P}}_t = 1$ implies that the entire service is hosted at the edge in time-slot t, and $\rho^{\mathcal{P}}_t = 0$ implies that the service is not hosted at the edge in time-slot t. The total cost incurred in time-slot t by policy $\mathcal{P}$ denoted by $\mathcal{C}^{\mathcal{P}}_t(r_t)$ is the sum of the rent, service, and fetch costs. It follows that

$$\mathcal{C}^{\mathcal{P}}_t(r_t) = c\rho^{\mathcal{P}}_t + g(\rho^{\mathcal{P}}_t) r_t + M(\rho^{\mathcal{P}}_t - \rho^{\mathcal{P}}_{t-1})^+. \quad (1)$$

Let $r = \{r_t\}_{t \geq 1}$ denote the request arrival sequence and $\mathcal{C}^{\mathcal{P}}(T, r)$ denote the cumulative cost incurred by policy $\mathcal{P}$ in time-slots 1 to T. It follows that

$$\mathcal{C}^{\mathcal{P}}(T, r) = \sum_{t=1}^{T} \mathcal{C}^{\mathcal{P}}_t(r_t).$$

Performance metrics: We consider regret as a performance metric.

For i.i.d. stochastic arrivals, the regret of a policy $\mathcal{P}$, denoted by $\mathcal{R}^{\mathcal{P}}_S(T)$, is defined as the difference in the total expected cost incurred by the policy and the optimal static hosting policy. The optimal static hosting policy makes a hosting decision at $t = 1$ using the knowledge of the statistics of the request arrival process, but not the entire sample-path. For ease of notation, we define $\mu = \mathbb{E}[r_t]$, $\mu_i = c\alpha_i + g(\alpha_i)\mu$. It follows that the expected cost incurred by the optimal static hosting policy in time-slots 1 to T is $\min_i\{c\alpha_i T + g(\alpha_i)\mu T + M\alpha_i\}$, and therefore,

$$\begin{aligned}\mathcal{R}^{\mathcal{P}}_S(T) &= \mathbb{E}_{\mathcal{P},r}[\mathcal{C}^{\mathcal{P}}(T, r)] - \min_i\{c\alpha_i T + g(\alpha_i)\mu T + M\alpha_i\} \\ &= \mathbb{E}_{\mathcal{P},r}[\mathcal{C}^{\mathcal{P}}(T, r)] - \min_i\{\mu_i T + M\alpha_i\}.\end{aligned}$$

For adversarial arrivals, the regret of a policy $\mathcal{P}$, denoted by $\mathcal{R}^{\mathcal{P}}_A(T)$, is defined as the worst case difference in the total expected cost incurred by the policy and the optimal static hosting policy. The expectation is taken over the policy. It follows that for any request sequence $r \in \mathbb{R}$, the total cost incurred by the optimal static hosting policy in time-slots 1 to T is $\min_i\{c\alpha_i T + g(\alpha_i)R_T + M\alpha_i\}$, and therefore,

$$\mathcal{R}^{\mathcal{P}}_A(T, r) = \mathbb{E}_{\mathcal{P}}[\mathcal{C}^{\mathcal{P}}(T, r)] - \min_i\{c\alpha_i T + g(\alpha_i)R_T + M\alpha_i\}$$

$$\mathcal{R}^{\mathcal{P}}_A(T) = \sup_{r \in \mathcal{R}} \left(\mathcal{R}^{\mathcal{P}}_A(T, r)\right).$$

Goal: The goal is to design online hosting policies with provable performance guarantees with respect to regret.

We will define some notation which will be useful in our analysis. Let $\mathbf{s} = [0, \alpha_1, \alpha_2, \cdots, 1]'$, $\mathbf{f} = [1, g(\alpha_1), g(\alpha_2), \cdots, 0]'$, and $\boldsymbol{\rho}^{\mathcal{P}}_t \in \mathcal{X}$, where $\mathcal{X}$ is the collection of all possible one hot vectors in $\{0,1\}^K$. The position of 1 in the one hot vector $\boldsymbol{\rho}^{\mathcal{P}}_t$ represents the level of service hosted at the edge, that is $\rho^{\mathcal{P}}_t = \langle \boldsymbol{\rho}^{\mathcal{P}}_t, \mathbf{s} \rangle$. Note that

[2]The entire request sequence is assumed to be fixed apriori.

$|\mathcal{X}| = K$. The total cost in slot t given in (1) can be rewritten using the above notations as follows,

$$\begin{aligned}\mathcal{C}_t^{\mathcal{P}}(r_t) &= c\langle \boldsymbol{\rho}_t^{\mathcal{P}}, \boldsymbol{s}\rangle + r_t\langle \boldsymbol{\rho}_t^{\mathcal{P}}, \boldsymbol{f}\rangle + M(\rho_t^{\mathcal{P}} - \rho_{t-1}^{\mathcal{P}})^+, \\ &= \langle \boldsymbol{\rho}_t^{\mathcal{P}}, \boldsymbol{\theta}_t\rangle + M(\rho_t^{\mathcal{P}} - \rho_{t-1}^{\mathcal{P}})^+,\end{aligned}$$

where $\boldsymbol{\theta}_t = c\boldsymbol{s} + r_t\boldsymbol{f}$.

For the ease of notation we define $\Delta_{ij} = \mu_i - \mu_j$, $i^\star = \arg\min_i \mu_i$, $\Delta_i = \mu_i - \mu_{i^\star}$, $\Delta_{\min} = \min_i \Delta_i$, $\Delta_{\max} = \max_i \Delta_i$.

III. Policies

Our Policy: Our policy called Wait then Follow the Perturbed Leader (W-FTPL) is a variant of the popular FTPL policy. A formal definition of FTPL is given in Algorithm 1. FTPL is a randomized policy and is known to perform well for the caching problem, in fact achieving order-wise optimal regret for adversarial arrivals [7]. Under FTPL, in any time slot t, for each i, FTPL considers the cumulative cost that would have been incurred had the system used α_i hosting fraction in the entire duration [1, t -1], and then perturbs this by an appropriately scaled version of a sample of a standard Gaussian random variable. FTPL then hosts the fraction of service for which the perturbed cost is minimum in that slot.

Algorithm 1: Follow The Perturbed Leader (FTPL)

Input: c, $\{\eta_t\}_{t\geq 1}$, request sequece$\{r_t\}_{t\geq 1}$
Set $\boldsymbol{\Theta}_1 \leftarrow \mathbf{0}$
Sample $\boldsymbol{\gamma} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
for $t = 1$ *to* T **do**
  host $\boldsymbol{\rho}_t \in \arg\min_{\boldsymbol{\rho}\in\mathcal{X}} \langle \boldsymbol{\rho}, \boldsymbol{\Theta}_t + \eta_t\boldsymbol{\gamma}\rangle$
  Update $\boldsymbol{\Theta}_{t+1} = \boldsymbol{\Theta}_t + \boldsymbol{\theta}_t$

The key idea behind the W-FTPL policy is to not host the service for an initial wait period. This is to reduce the number of fetches made initially when estimate of the arrivals is noisy. The duration of this wait period is not fixed apriori and is a function of the arrival pattern seen till that time. Following the wait period, W-FTPL mimics the FTPL policy. Refer to Algorithm 2 for a formal definition of W-FTPL. Let T_s be the time slot after which W-FTPL starts acting as FTPL. Formally we define $T_s = \min\{t : t < \frac{(\max_{i\neq j}(\Theta_{t+1,i} - \Theta_{t+1,j}))^2}{\kappa^2\beta(\log M)^{1+\delta}}\}$, where $\beta > 1$, $\delta > 0$ are constants .

Retro-Renting (RR) [3]: The RR policy is a deterministic hosting policy and it either hosts the complete service or hosts nothing in a slot. The key idea behind this policy is to use recent arrival patterns to make hosting decisions. We refer the reader to Algorithm 1 in [3] for a formal definition of the RR policy. The performance of RR with respect to the competitive ratio was analyzed in [3].

$\alpha-$RR[2] policy is modified version of RR where one partial level of hosting the service is allowed. The formal definition of $\alpha-$RR is given in Algorithm 3 which is Algorithm 1 in [2]. Similar to RR the key idea behind $\alpha-$RR policy is to use recent arrival patterns to make hosting decisions. We refer $\alpha-$RR also as RR in this paper. Based on the levels of hosting allowed it will be clear either it is $\alpha-$RR or RR.

Algorithm 2: Wait then Follow The Perturbed Leader (W-FTPL)

Input: c, M, $\{\eta_t\}_{t\geq 1}$, $\{r_t\}_{t\geq 1}$, β, δ, κ
Set $\boldsymbol{\Theta_1} \leftarrow \mathbf{0}$, wait=1
Sample $\boldsymbol{\gamma} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
for $t = 1$ *to* T **do**
  if *wait* $= 1$ **then**
    $\rho_t = 0$
  else
    host $\boldsymbol{\rho}_t \in \arg\min_{\boldsymbol{\rho}\in\mathcal{X}} \langle \boldsymbol{\rho}, \boldsymbol{\Theta}_t + \eta_t\boldsymbol{\gamma}\rangle$
  Update $\boldsymbol{\Theta}_{t+1} = \boldsymbol{\Theta}_t + \boldsymbol{\theta}_t$
  wait $= \min\{\text{wait}, \mathbb{1}_{t > \frac{(\max_{i\neq j}(\Theta_{t+1,i} - \Theta_{t+1,j}))^2}{\kappa^2\beta(\log M)^{1+\delta}}}\}$

Algorithm 3: α-RetroRenting (α-RR)

Input: Fetch cost M, partial hosting level α_2, latency cost under partial hosting $g(\alpha_2)$, rent cost c, request arrival sequence $\{x_l\}_{l\geq 0}^t$
Output: service hosting strategy r_{t+1}, $t > 0$
Initialize: $r_1 = t_{\text{recent}} = 0$
for ***each*** *time-slot* t **do**
  $I_t = (M,\, g(\alpha_2),\, t,\, t_{\text{recent}},\, c,\, \{x_l\}_{l\geq t_{\text{recent}}}^t)$
  $R_0^{(\tau_0)} = [\underbrace{r_t, r_t, \ldots, r_t}_{\tau_0 - t_{\text{recent}}}, \underbrace{0, 0, \ldots, 0}_{t-\tau_0}]$
  $R_\alpha^{(\tau_\alpha)} = [\underbrace{r_t, r_t, \ldots, r_t}_{\tau_\alpha - t_{\text{recent}}}, \underbrace{\alpha_2, \alpha_2, \ldots, \alpha_2}_{t-\tau_\alpha}]$
  $R_1^{(\tau_1)} = [\underbrace{r_t, r_t, \ldots, r_t}_{\tau_1 - t_{\text{recent}}}, \underbrace{1, 1, \ldots, 1}_{t-\tau_1}]$
  $\text{minCost}(0) = \min_{\tau_0\in(t_{\text{recent}},t)} \texttt{totalCost}(R_0^{(\tau_0)}, I_t)$
  $\text{minCost}(\alpha_2) = \min_{\tau_\alpha\in(t_{\text{recent}},t)} \texttt{totalCost}(R_\alpha^{(\tau_\alpha)}, I_t)$
  $\text{minCost}(1) = \min_{\tau_1\in(t_{\text{recent}},t)} \texttt{totalCost}(R_1^{(\tau_1)}, I_t)$
  $r_{t+1} = \arg\min_{i\in\{0,\alpha_2,1\}} \text{minCost}(i)$
  if $r_{t+1} \neq r_t$ **then**
    $t_{\text{recent}} = t$
  end
end
Function `totalCost`(R, I_t)**:**
  $g(0) = 1$, $g(1) = 0$;
  cost = $R(1) \times c + x_1 \times g(R(1))$;
  for $j \leftarrow 2$ **to** $t - t_{recent}$ **do**
    cost = cost $+R(j) \times c + x_j \times g(R(j))$
      $+M \times |R(j) - R(j-1)|$;
  end
  return cost;
end

IV. Main Results and Discussion

In this section, we state and discuss our key results.

A. Regret Analysis

Our first result characterizes the regret performance of the policies discussed in Section III and the fundamental limit on the performance of any online policy for adversarial arrivals.

Theorem 1 (Adversarial Arrivals): Let the arrivals be generated by an oblivious adversary under the constraint that at most κ request arrives in each time-slot, then,

(a) $\mathcal{R}_A^{\mathcal{P}}(T) = \Omega(\sqrt{T})$ for any online policy $\mathcal{P}$.

(b) $\mathcal{R}_A^{\text{RR}}(T) = \Omega(T)$, $\mathcal{R}_A^{\alpha-\text{RR}}(T) = \Omega(T)$.

(c) For $\eta_t = \alpha\sqrt{t}$, $\alpha > 0$,

$$\mathcal{R}_A^{\text{FTPL}}(T) \leq \sqrt{2T\log K}\left(\alpha + \frac{4\kappa^2}{\alpha}\right) + \frac{K^2 M(c+2\kappa)}{2\alpha\sqrt{\pi}}\sqrt{T+1}.$$

(d) For $\eta_t = \alpha\sqrt{t}$, $\alpha > 0$, $\beta > 1$, $\delta > 0$,

$$\mathcal{R}_A^{\text{W-FTPL}}(T) \leq \sqrt{\beta T(\log M)^{1+\delta}} + \sqrt{2T\log K}\left(\alpha + \frac{4\kappa^2}{\alpha}\right) + \frac{K^2 M(c+2\kappa)}{2\alpha\sqrt{\pi}}\sqrt{T+1}.$$

The key takeaways from Theorem 1 is that RR has linear regret and FTPL, W-FTPL have order-optimal regret, i.e., $\text{O}(\sqrt{T})$ with respect to the time horizon under adversarial arrivals. The proof of Theorem 1 is in Section VI.

Our second result characterizes the regret performance of the policies discussed in Section III for i.i.d. stochastic arrivals.

Theorem 2 (Stochastic Arrivals): Let the arrivals in each time-slot be i.i.d. stochastic with mean μ and $\Delta_{\min} \neq 0$. Then we have

(a) $\mathcal{R}_S^{\text{RR}}(T) = \Omega(T)$, $\mathcal{R}_S^{\alpha-\text{RR}}(T) = \Omega(T)$.

(b) For $\eta_t = \alpha\sqrt{t-1}$, $\alpha > 0$,

$$\mathcal{R}_S^{\text{FTPL}}(T) \leq \left(\sqrt{2\log K} + \frac{2\sqrt{2h_1}\log K}{\Delta_{\min}}\right)\left(\alpha + \frac{4\kappa^2}{\alpha}\right) + \frac{16\alpha^2 + 4\kappa^2}{\Delta_{\min}} + (16\alpha^2 + 3\kappa^2)\frac{MK^2}{\Delta_{\min}^2},$$

where $h_1 = 4\max\{8\alpha^2, \kappa^2\}$.

(c) For $\eta_t = \alpha\sqrt{t-1}$, $\alpha > 0$, $\beta > 1$, $\delta > 0$, and M large enough

$$\mathcal{R}_S^{\text{W-FTPL}}(T) \leq \frac{4\beta\kappa^2(\log M)^{1+\delta}}{\Delta_{\min}^2} + (16\alpha^2 + 4\kappa^2) \times \left(\frac{1}{\Delta_{\min}^2} + \frac{\beta\kappa^2}{\Delta_{\min}^2\Delta_{\max}^2} + \sum_{i\neq i^\star}\frac{1}{\Delta_i}\right).$$

Remark 1: For the W-FTPL policy, if $\delta = 0$ then for large value of β, we have an upper bound on regret that scales logarithmically with M. For $\delta > 0$, for $\beta > 1$, we have an upper bound on regret that scales as $(\log M)^{1+\delta}$.

The key take away from Theorem 2 is that RR has linear regret w.r.t. time and FTPL, W-FTPL has constant regret w.r.t. time. Also, the regret of W-FTPL is proportional to $(\log M)^{1+\delta}$ for $\delta \geq 0$, where as for FTPL, regret scales linearly with M.

In Section V we validate the results stated in this section through simulations.

V. Simulations

In this section we compare RR, FTPL and W-FTPL policies via simulations using synthetic, stochastic arrivals.

A. Synthetic arrivals

In Fig. 1 we plot the regret of different policies as a function of time horizon T. The parameters considered to generate arrivals are $c = 0.1$, $M = 50$, $\kappa = 5$. The request arrival considered is similar to the one used in proof of Theorem 1(b) i.e., we divide time into frames and each frame stars with $\lceil \frac{M}{\kappa - c} \rceil$ slots of κ requests followed by $\lceil \frac{M}{c} \rceil$ slots of zero requests. We consider 100 such frames for this experiment. We observe that RR has linear regret in this case which agrees with Theorem 1.

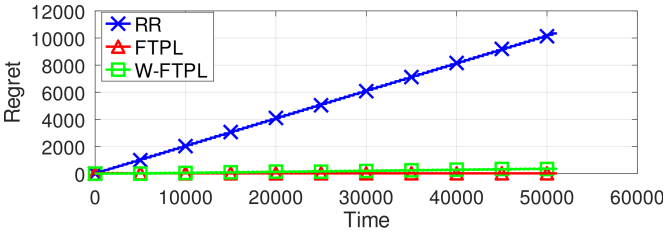

Fig. 1: Regret as a function of Time horizon (T)

B. Stochastic arrivals

For the stochastic case we consider arrivals in each slot to be Bernoulli with parameter μ. We will restrict attention to the case where there is only one partial level of hosting, i.e., $\rho_t^{\mathcal{P}} \in \{0, \alpha_2, 1\}$. We consider $\alpha_2 = 0.5$ and $g(\alpha_2) = 0.45$. In Fig.2 and Fig.3, we consider $\mu = 0.4$, $c = 0.45$ and $\beta = 6$, $\eta_t = 0.1\sqrt{t}$ for FTPL, W-FTPL policies. All the results plotted are averaged over 50 independent experiments. In Fig.2, we fix $M = 5$ and compare the regret performance of the policies as a function of time. We observe that W-FTPL performs better than other polices and RR has linear regret which agrees with Theorem 2. In Fig.3, we compare the performance of the policies with respect to M for $T = 10^4$. We observe that W-FTPL performs better than other policies and the main difference between the cost of policies is due to fetch cost. Also note that W-FTPL has better dependence on M as compared to FTPL which agrees with Theorem 2 . In Fig.4, we compare the performance of the policies by varying rent cost c for $M = 500$, $\mu = 0.45$, and $T = 10^4$. Again we note that W-FTPL outperforms RR and FTPL. As c gets closer to μ the FTPL policy does multiple switches and as a result the regret increases when c is close to μ. The increase in switch cost is due to decrease in Δ which follows from Theorem 2.

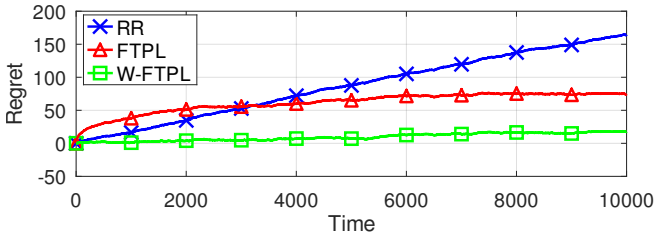

Fig. 2: Regret as a function of time

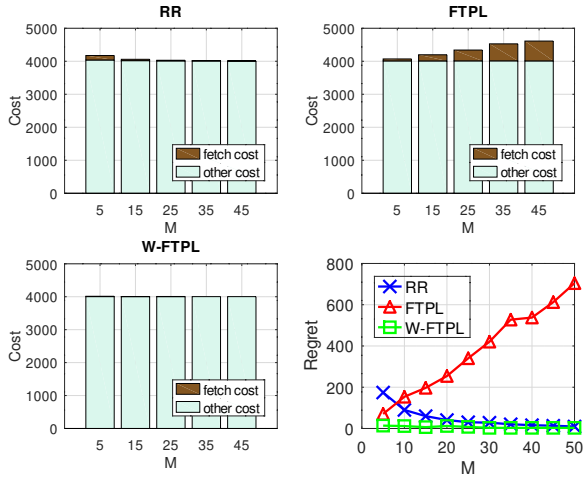

Fig. 3: Cost, Regret as a function of fetch cost (M)

Our simulation results thus indicate that FTPL and W-FTPL outperform RR for i.i.d. stochastic arrivals. Additionally, RR can have linear regret with respect to time, while FTPL and W-FTPL have sub-linear regret with respect to time for all arrival processes considered.

VI. Proof of Theorems

Due to space constraints, we present the proofs of some of our results.

A. Proof of Theorem 1(a)

We use the following lemmas to prove Theorem 1(a).

Lemma 1: If X is a random variable and $\mathbb{E}[f_1(X)] = \mathbb{E}[f_2(X)] = m$, then

$$m - \mathbb{E}[\min\{f_1(X), f_2(X)\}] = \frac{1}{2}\mathbb{E}[|f_1(X) - f_2(X)|].$$

Lemma 2: If $X \sim \text{Bin}(T, p)$ then $\frac{1}{2}\mathbb{E}[|X - Tp|] = \Omega(T)$.

Proof of Theorem 1(a): Recall that regret of any policy is given by

$$\mathcal{R}_A^{\mathcal{P}}(T, r) = \sum_{t=1}^{T} c\rho_t + g(\rho_t) r_t + M(\rho_t - \rho_{t-1})^+ - \min_i \{c\alpha_i T + g(\alpha_i) R_T + M\alpha_i\}.$$

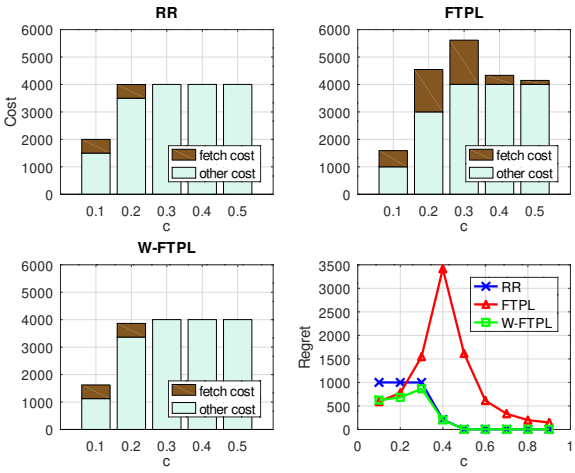

Fig. 4: Cost, Regret as a function of rent cost (c)

Since $\sup_{r\in\mathcal{R}} \mathcal{R}_A^{\mathcal{P}}(T, r) \geq \mathbb{E}_r[\mathcal{R}_A^{\mathcal{P}}(T, r)]$ where the expectation is taken over request sequences in $\mathcal{R}$. We lower bound $\mathbb{E}_r[\mathcal{R}_A^{\mathcal{P}}(T, r)]$ to get the lower bound on $\mathcal{R}_A^{\mathcal{P}}(T)$.

$$\begin{aligned}\mathbb{E}_r[\mathcal{R}_A^{\mathcal{P}}(T, r)] &= \sum_{t=1}^{T} c\rho_t + g(\rho_t)\mathbb{E}_r[r_t] + M(\rho_t - \rho_{t-1})^+ \\ &\quad - \mathbb{E}_r[\min_i \{c\alpha_i T + g(\alpha_i) R_T + M\alpha_i\}] \\ &\geq \sum_{t=1}^{T} c\rho_t + g(\rho_t)\mathbb{E}_r[r_t] \\ &\quad - \mathbb{E}_r[\min_i \{c\alpha_i T + g(\alpha_i) R_T\}] - M.\end{aligned}$$

Let $\ell = \arg\min_{i\neq 1} \frac{\alpha_i}{1-g(\alpha_i)}$, $X \sim \text{Ber}(\frac{c\alpha_\ell}{\kappa(1-g(\alpha_\ell))})$, and $r_t = \kappa X$, therefore $\mathbb{E}_r[r_t] = \frac{c\alpha_\ell}{1-g(\alpha_\ell)}$. We consider $c < \kappa$ and $\alpha_i + g(\alpha_i) \leq 1$ for all $1 \leq i \leq K$ therefore $\frac{c\alpha_\ell}{\kappa(1-g(\alpha_\ell))} < 1$ and X is a valid Bernoulli random variable.

$$\begin{aligned}\mathbb{E}_r[\mathcal{R}_A^{\mathcal{P}}(T, r)] &\geq \sum_{t=1}^{T} c\rho_t + g(\rho_t)\frac{c\alpha_\ell}{1-g(\alpha_\ell)} \\ &\quad - \mathbb{E}_r[\min_i \{c\alpha_i T + g(\alpha_i) R_T\}] - M\end{aligned}$$

$$\begin{aligned}&= \sum_{t=1}^{T}(1-g(\rho_t))\frac{c\rho_t}{(1-g(\rho_t))}\mathbb{1}_{\rho_t\neq 0} + g(\rho_t)\frac{c\alpha_\ell}{1-g(\alpha_\ell)} \\ &\quad - \mathbb{E}_r[\min_i \{c\alpha_i T + g(\alpha_i) R_T\}] - M \\ &\overset{(a)}{\geq} \frac{cT\alpha_\ell}{1-g(\alpha_\ell)} - \mathbb{E}_r[\min\{R_T, cT\alpha_\ell + g(\alpha_\ell) R_T\}] - M \\ &\overset{(b)}{\geq} \frac{1}{2}\mathbb{E}_r[|R_T - cT\alpha_\ell - g(\alpha_\ell) R_T|] - M \\ &= \frac{\kappa(1-g(\alpha_\ell))}{2}\mathbb{E}_r\left[\left|\frac{R_T}{\kappa} - \frac{Tc\alpha_\ell}{\kappa(1-g(\alpha_\ell))}\right|\right] - M \\ &\overset{(c)}{=} \Omega(\sqrt{T}),\end{aligned}$$

where (a) follows from definition of ℓ, (b) follows from Lemma 1, (c) follows from Lemma 2. ■

B. Proof of Theorem 1(b)

We use the following lemmas to prove Theorem 1(b).

Lemma 3: RR does not fetch the service for at least $\lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil$ slots after eviction, where $\alpha_{i'} = \arg\min_{\alpha_i \neq 0} \frac{M\alpha_i}{\kappa - c\alpha_i - \kappa g(\alpha_i)}$.

Lemma 4: Once fetched, RR hosts the service for at least $\lceil \frac{M}{c} \rceil$ slots before it evicts the service.

Proof of Theorem 1(b): We split the entire time duration T into frames of size $t_f + t_e$ and consider the arrival sequence in each frame as $r_t = \kappa$ for $1 \leq t \leq t_f$ and $r_t = 0$ for $t_f + 1 \leq t \leq t_f + t_e$, where $\alpha_{i'} = \arg\min_{\alpha_i \neq 0} \frac{M\alpha_i}{\kappa - c\alpha_i - \kappa g(\alpha_i)}$, $t_f = \lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil$ and $t_e = \lceil \frac{M}{c} \rceil$.

From Lemma 3 we know that RR does not fetch any fraction of service till time t_f. By definition of α' and arrivals sequence, for $t \leq t_f$, $R_t = kt$. We have $c\alpha_{i'} t_f + g(\alpha_{i'}) R_{t_f} + M\alpha_{i'} < R_{t_f}$, which is the condition to fetch $\alpha_{i'}$ in $t = t_f + 1$. By using Lemma 4 we know that RR does not evict the service for $t_f + 1 \leq t \leq t_f + t_e$. At $t = t_f + t_e + 1$ RR evicts the service since condition to evict the service is satisfied i.e., $c\alpha_{i'} t_e + g(\alpha_{i'}) \sum_{t=t_f+1}^{t_f+t_e} r_t < M\alpha_{i'} + \sum_{t=t_f+1}^{t_f+t_e} r_t$.

So amongst the first $t_f + t_e$ timeslots, the RR policy hosts for t_e slots and does not host for t_f slots. If the optimal static policy is to host $\alpha_{i^*} \neq 0$ fraction of service then we have regret at least $(\kappa - c\alpha_{i'} + g(\alpha_{i'})\kappa) t_f = (\kappa - c\alpha_{i'} + g(\alpha_{i'})\kappa) \lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil > 0$ till time $t_f + t_e$, which is a constant and does not depend on T. If optimal is not to host any fraction of service then we have regret of $c\alpha_{i'} t_e > 0$ till $t_f + t_e$, which is a constant and does not depend on T. Either the optimal static policy is to host $\alpha_{i^*} \neq 0$ or $\alpha_{i^*} = 0$ there will always be finite regret in the frame duration say $d (> 0)$ and d is independent of T. Since we split the entire time duration T into frames of size $t_f + t_e = \lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil + \lceil \frac{M}{c} \rceil$ and the request sequence are repeated in each frame we get linear regret.

$$\begin{aligned}
\mathcal{R}_A^{\text{RR}}(T) &\geq \sum_{f=1}^{\lfloor T/(t_f+t_e) \rfloor} d \\
&\geq \lfloor T/(t_f + t_e) \rfloor d \\
&\geq d \left(\frac{T}{\frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} + \frac{M}{c} + 2} - 1 \right) = \Omega(T).
\end{aligned}$$

■

C. Proof of Theorem 1(c)

The optimal policy only fetches the optimal fraction of service to be hosted once. Therefore,

$$\begin{aligned}
&\mathcal{R}_A^{\text{FTPL}}(T, r) \\
&= \mathbb{E}[\mathcal{C}^{\text{FTPL}}(T, r)] - \min_i \{c\alpha_i T + g(\alpha_i) R_T + M\alpha_i\} \\
&\leq \mathbb{E}[\mathcal{C}^{\text{FTPL}}(T, r)] - \min_i \{c\alpha_i T + g(\alpha_i) R_T\} \qquad (2)
\end{aligned}$$

Since FTPL policy does not consider fetch cost M while making decisions we can decouple the fetch cost and the non fetch cost incurred by the FTPL policy. Therefore we can also decouple the expected regret incurred by the FTPL into expected regret without fetch cost ($M = 0$) and fetch cost incurred by FTPL to bound (2).

We first bound expected regret of the FTPL policy with fetch cost $M = 0$ and add the expected fetch cost to get the final regret bound.

Lemma 5: The regret for FTPL policy with non decreasing learning rate $\{\eta_t\}_{t=1}^T$ and $M = 0$ is given by

$$\mathcal{R}_A^{\text{FTPL}}(T) \leq \sqrt{2 \log K} \left(\eta_T + \kappa^2 \sum_{t=1}^{T} \frac{1}{\eta_t} \right).$$

Now we bound the expected fetch cost incurred by FTPL policy.

Lemma 6: The expected fetch cost under FTPL policy upto time T denoted by $\mathbb{E}[\mathcal{C}_F^{\text{FTPL}}(T)]$ with learning rate $\eta_t = \alpha\sqrt{t}$ is bounded as follows

$$\mathbb{E}[\mathcal{C}_F^{\text{FTPL}}(T)] \leq \frac{MK^2(c + 2\kappa)}{2\alpha\sqrt{\pi}} \sqrt{T + 1}.$$

Proof of Theorem 1(c): By using Lemma 5 and 6 we get,

$$\begin{aligned}
\mathcal{R}_A^{\text{FTPL}}(T) \leq &\sqrt{2 \log K} \left(\alpha\sqrt{T} + 2\kappa^2 \sum_{t=1}^{T} \frac{1}{\alpha\sqrt{t}} \right) \\
&+ \frac{K^2 M(c + 2\kappa)}{2\alpha\sqrt{\pi}} \sqrt{T + 1},
\end{aligned}$$

and the result follows. ■

D. Proof of Theorem 1(d)

Proof of Theorem 1(d): Recall that $T_s = \min\{t : t < \frac{(\max_{i \neq j} (\Theta_{t+1,i} - \Theta_{t+1,j}))^2}{\kappa^2 \beta (\log M)^{1+\delta}}\}$. For arrival sequences with requests arriving in time-slots 1 to T, we have two possible cases, namely, $T_S \geq T$ and $T_s < T$. We consider each case to bound the regret of W-FTPL policy.

Case I ($T_s \geq T$): Let $r^{(1)}$ be a request sequence chosen by adversary such that $T_s \geq T$, then

$$\begin{aligned}
&|\max_{i \neq j} \Theta_{T+1,j} - \Theta_{T+1,i}| < \kappa \sqrt{\beta T (\log M)^{1+\delta}} \\
&\implies \mathcal{R}_A^{\text{W-FTPL}}(T) < \kappa \sqrt{\beta T (\log M)^{1+\delta}}
\end{aligned}$$

Case II ($T_s < T$): Let $r^{(2)}$ be a request sequence chosen by adversary such that $T_s < T$, then by using case I we can bound

$$\begin{aligned}
&|\max_{i \neq j} \Theta_{T_s+1,i} - \Theta_{T_s+1,j}| < \kappa \sqrt{\beta T_s (\log M)^{1+\delta}} \\
&\implies \mathcal{R}_A^{\text{W-FTPL}}(T_s) < \kappa \sqrt{\beta T (\log M)^{1+\delta}}.
\end{aligned}$$

Thus combining both cases we get upper bound on regret in wait phase as

$$\mathcal{R}_A^{\text{W-FTPL}}(T_s) < \kappa \sqrt{\beta T (\log M)^{1+\delta}}. \qquad (3)$$

Note that W-FTPL policy follows FTPL policy after its waiting time i.e., from time $T_s + 1$. Therefore W-FTPL and FTPL take same decisions and have cost from time $T_s + 1$ to T. Thus the regret in a slot is also same for W-FTPL and FTPL from time $T_s + 1$ to T. If $T_s \geq T$ then only wait phase

will be there and regret in FTPL phase is considered to be zero. We denote regret from time t_1 to t_2 as $\mathcal{R}_A^{\mathcal{P}}(t_1 : t_2)$.

$$\begin{aligned}\mathcal{R}_A^{\text{W-FTPL}}(T) &= \mathcal{R}_A^{\text{W-FTPL}}(T_s) + \mathcal{R}_A^{\text{W-FTPL}}(T_s : T) \\ &= \mathcal{R}_A^{\text{W-FTPL}}(T_s) + \mathcal{R}_A^{\text{FTPL}}(T_s : T) \\ &\leq \mathcal{R}_A^{\text{W-FTPL}}(T_s) + \mathcal{R}_A^{\text{FTPL}}(1 : T) \qquad (4)\end{aligned}$$

By (3), (4) and Theorem 1(c) we get

$$\begin{aligned}\mathcal{R}_A^{\text{W-FTPL}}(T) \leq & \kappa\sqrt{\beta T(\log M)^{1+\delta}} + \frac{K^2 M(c+2\kappa)}{2\alpha\sqrt{\pi}}\sqrt{T+1} \\ & + \sqrt{2T\log K}\left(\alpha + \frac{4\kappa^2}{\alpha^2}\right).\end{aligned}$$

■

E. Proof of Theorem 2(a)

Proof of Theorem 2(a): From Lemma 4 we know that RR hosts the service if fetched for at least $\lceil \frac{M}{c} \rceil$ slots and from Lemma 3 we know once evicted RR takes at least $\lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil$ slots to fetch $\alpha_{i'}$ fraction of service. We divide the entire time T into frames of size $f_s = \lceil \frac{M}{c} \rceil + \lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil$ slots. Let us define an event $\mathcal{F}$ as the frame that starts with $\lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil$ slots with κ arrivals in each slot and followed by $\lceil \frac{M}{c} \rceil$ slots of zeros arrivals. Let the probability of event $\mathcal{F}$ occurring be p. Since requests are i.i.d probability of getting κ requests or zero requests in a slot are independent of time horizon T which implies probability of observing frame $\mathcal{F}$ is also independent of T. Conditioned on event $\mathcal{F}$, if optimal static policy is to host $\alpha_{i^*} \neq 0$ fraction of service then we have regret at least $(\kappa - c\alpha + g(\alpha)\kappa)\lceil \frac{M\alpha_{i'}}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} \rceil > 0$ in a frame. If optimal is not to host any fraction of service then we have regret of $c\alpha_{i'}\lceil \frac{M}{c} \rceil > 0$ in a frame. Therefore conditioned on event $\mathcal{F}$ RR always have a finite nonzero regret say $d(> 0)$ which is independent of T. Therefore,

$$\begin{aligned}\mathcal{R}_S^{\text{RR}}(T) &= \mathbb{E}_r\left[\sum_{f=1}^{\lfloor T/f_s \rfloor} d\mathbb{1}(\text{event } \mathcal{F} \text{ occured})\right] \\ &\geq \left(\frac{T}{f_s} - 1\right) pd \\ &\geq \left(\frac{T}{\frac{M}{\kappa - c\alpha_{i'} - g(\alpha_{i'})\kappa} + \frac{M}{c} + 2} - 1\right) pd \\ &= \Omega(T).\end{aligned}$$

So even in the stochastic case RR observes linear regret. ■

F. Proof of Theorem 2(b)

We use the following lemmas to prove Theorem 2(b).

Lemma 7: For $M = 0$ we bound the Regret of FTPL policy as follows

$$\mathcal{R}_S^{\text{FTPL}}(T) \leq \sum_{i \neq i^\star} \frac{16\alpha^2 + 3\kappa^2}{\Delta_i}.$$

We improve the bound on regret of FTPL in Lemma 7 to get better dependence on number of storage levels we can have.

Lemma 8: For $M = 0$ we bound the Regret of FTPL policy as follows

$$\begin{aligned}\mathcal{R}_S^{\text{FTPL}}(T) \leq & \left(\sqrt{2\log K} + \frac{2\sqrt{2h_1}\log K}{\Delta_{\min}}\right)\left(\alpha + \frac{4\kappa^2}{\alpha}\right) \\ & + \frac{16\alpha^2 + 4\kappa^2}{\Delta_{\min}},\end{aligned}$$

where $h_1 = 2\max\{16\alpha^2, 2\kappa^2\}$

Lemma 9: Fetch cost under FTPL policy with learning rate $\eta_t = \alpha\sqrt{t-1}$ is bounded as follows

$$\mathbb{E}[\mathcal{C}_f^{\text{FTPL}}(T)] \leq MK^2\frac{16\alpha^2 + 2\kappa^2}{\Delta_{\min}^2}.$$

Proof of Theorem 2(b): Note that FTPL policy does not consider fetch cost M while taking the decisions. By using Lemma 8, 9 we get the result stated. ■

G. Proof of Theorem 2(c)

We use the following lemmas to prove Theorem 2(c).

Lemma 10: Under FTPL policy with learning rate $\eta_t = \alpha\sqrt{t-1}$, $\alpha_i < \alpha_j$ we have

$$\begin{aligned}\mathbb{P}(\rho_t = \alpha_i, \rho_{t+1} = \alpha_j) \leq & \exp\left(-\frac{(t-1)\Delta_{\min}^2}{16\alpha^2}\right) \\ & + \exp\left(\frac{-\Delta_{\min}^2(t-1)}{2\kappa^2}\right).\end{aligned}$$

Lemma 11: Under the W-FTPL policy, $T_s > \frac{(\sqrt{\beta}-1)^2(\log M)^{1+\delta}}{\Delta_{\max}^2}$ with probability at least $1 - \frac{(\sqrt{\beta}-1)^2(\log M)^{1+\delta}}{M^{2(\log M)^\delta}\Delta_{\max}^2}$.

Lemma 12: Under the W-FTPL policy

$$\mathbb{E}[T_s] \leq 1 + \frac{4\beta\kappa^2(\log M)^{1+\delta}}{\Delta_{\min}^2} + \frac{1}{M^{2\beta}}\frac{2\kappa^2}{\Delta_{\min}^2}.$$

Proof of Theorem 2(c): Under W-FTPL,

$$\begin{aligned}\mathbb{E}_r[\mathcal{C}^{\text{W-FTPL}}(T)] = & \mathbb{E}_r[T_s] + \mathbb{E}_r\left[\sum_{t=T_s+1}^{T}(c\rho_t + g(\rho_t)r_t)\right] \\ & + M\mathbb{E}_r\left[\sum_{t=T_s+1}^{T}\mathbb{P}(\rho_t > \rho_{t-1})\right].\end{aligned}$$

By the using definition of regret we get,

$$\begin{aligned}\mathcal{R}_S^{\text{W-FTPL}}(T) \leq & \mathbb{E}[T_s] + \sum_{t=1}^{T}(c\rho_t + \mu g(\rho_t)) - \mu_{i^\star} \\ & + M\mathbb{E}_r\left[\sum_{t=T_s+1}^{T}\mathbb{P}(\rho_t > \rho_{t-1})\right].\end{aligned}$$

By using the results of Lemma 7, 9 we get

$$\begin{aligned}\mathcal{R}_S^{\text{W-FTPL}}(T) \leq & \mathbb{E}[T_s] + \sum_{i \neq i^\star}\frac{16\alpha^2 + 3\kappa^2}{\Delta_i} \\ & + M\mathbb{E}_r\left[\sum_{t=T_s+1}^{T}\mathbb{P}(\rho_t > \rho_{t-1})\right].\end{aligned}$$

Let $\mathbb{E}_r[\mathcal{C}_f] = M\mathbb{E}_r\left[\sum_{t=T_s+1}^{T}\mathbb{P}(\rho_t > \rho_{t-1})\right]$, then

$$\begin{aligned}
\mathbb{E}_r[\mathcal{C}_f] =& \mathbb{E}_r[\mathcal{C}_f|T_s \leq T_0]\mathbb{P}(T_s \leq T_0) \\
&+ \mathbb{E}_r[\mathcal{C}_f|T_s > T_0]\mathbb{P}(T_s > T_0) \\
\leq& \mathbb{E}[\mathcal{C}_f|T_s = 1]\mathbb{P}(T_s \leq T_0) + \mathbb{E}[\mathcal{C}_f|T_s = \lceil T_0 \rceil] \\
\overset{(a)}{\leq}& \mathbb{P}(T_s \leq T_0)\mathbb{E}[\mathcal{C}_F^{\text{FTPL}}] \\
&+ MK^2 \sum_{t=\lceil T_0 \rceil}^{T} \left(\exp\left(-\frac{\Delta_{\min}^2 t}{16\alpha^2}\right) + e^{-\Delta_{\min}^2 t/2\kappa^2}\right) \\
\leq& MK^2\mathbb{P}(T_s \leq T_0)\frac{16\alpha^2 + 3\kappa^2}{\Delta_{\min}^2} \\
&+ MK^2\frac{16\alpha^2}{\Delta_{\min}^2}\exp\left(-\frac{\Delta_{\min}^2 T_0}{16\alpha^2}\right) \\
&+ MK^2\frac{2\kappa^2}{\Delta_{\min}^2}\exp\left(-\frac{\Delta_{\min}^2 T_0}{2\kappa^2}\right).
\end{aligned}$$

Here, (a) is obtained by using Lemma 10. By considering $T_0 = \frac{(\sqrt{\beta}-1)^2\kappa^2(\log M)^{1+\delta}}{\Delta_{\max}^2}$ and using Lemma 11, 12 we get,

$$\begin{aligned}
&\mathcal{R}_S^{\text{W-FTPL}}(T) \\
\leq& 1 + \frac{4\beta\kappa^2(\log M)^{1+\delta}}{\Delta_{\min}^2} + \frac{1}{M^{2\beta(\log M)^\delta}}\frac{2\kappa^2}{\Delta_{\min}^2} \\
&+ (16\alpha^2 + 3\kappa^2)\sum_{i \neq i^\star}\frac{1}{\Delta_i} \\
&+ MK^2\frac{(\sqrt{\beta}-1)^2\kappa^2(\log M)^{1+\delta}}{\Delta_{\max}^2 M^{2(\log M)^\delta}}\frac{16\alpha^2 + 3\kappa^2}{\Delta_{\min}^2} \\
&+ \left(\frac{16\alpha^2 MK^2}{M\frac{(\sqrt{\beta}-1)^2\kappa^2\Delta_{\min}^2(\log M)^\delta}{16\alpha^2\Delta_{\max}^2}} + \frac{2MK^2\kappa^2}{M\frac{(\sqrt{\beta}-1)^2\Delta_{\min}^2(\log M)^\delta}{2\Delta_{\max}^2}}\right)\frac{1}{\Delta_{\min}^2}.
\end{aligned}$$

For large values of M we get the result stated. ■

VII. Conclusions

We study the problem of (partial) service hosting at the edge for both adversarial and stochastic arrivals with regret as the performance metric of interest. We show that the RR policy and its variant for partial hosting are strictly sub-optimal for both adversarial and stochastic arrivals with linear regret with respect to time. We further show that the widely studied FTPL policy has order-optimal regret with respect to time for both adversarial and stochastic arrivals. One shortcoming of the FTPL policy is that its performance can deteriorate in the case where the cost of fetching the service to host at the edge is high. To address this limitation, we propose a variant of FTPL called Wait-then-FTPL and show that in addition to having order-optimal regret with respect to time for both adversarial and stochastic arrivals, W-FTPL outperforms FTPL when the cost of fetching is high.

References

[1] R. S. Prakash, N. Karamchandani, V. Kavitha, and S. Moharir, "Partial service caching at the edge," in *2020 18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*. IEEE, 2020, pp. 1–8.

[2] V. S. C. L. Narayana, M. Agarwala, R. S. Prakash, N. Karamchandani, and S. Moharir, "Online partial service hosting at the edge," *CoRR*, vol. abs/2103.00555, 2021. [Online]. Available: https://arxiv.org/abs/2103.00555

[3] V. C. L. Narayana, S. Moharir, and N. Karamchandani, "On renting edge resources for service hosting," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, vol. 6, no. 2, pp. 1–30, 2021.

[4] T. Zhao, I.-H. Hou, S. Wang, and K. Chan, "Red/led: An asymptotically optimal and scalable online algorithm for service caching at the edge," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1857–1870, 2018.

[5] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22, 1985.

[6] G. S. Paschos, A. Destounis, L. Vigneri, and G. Iosifidis, "Learning to cache with no regrets," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 235–243.

[7] R. Bhattacharjee, S. Banerjee, and A. Sinha, "Fundamental limits on the regret of online network-caching," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 4, no. 2, pp. 1–31, 2020.

[8] T. S. Salem, G. Neglia, and S. Ioannidis, "No-regret caching via online mirror descent," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[9] S. Mukhopadhyay and A. Sinha, "Online caching with optimal switching regret," in *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2021, pp. 1546–1551.